Usuario:ManuelRomero/php/NewPHP/B2T1/Sintaxis

De WikiEducator
< Usuario:ManuelRomero

LENGUAJE PHP: **EL LENGUAJE EN GENERAL**

¡El servidor te responde

PHP Un lenguaje de script al lado del servidor

Sintaxis del lenguaje | Ejercicios | Práctica | Volver

TEMA 3: LENGUAJE PHP

Contenido

Show presentation

Introducción a PHP

Sección de introducción a PHP

- En esta sección veremos qué es el lenguaje php y para qué sirve

Qué es php

- ***PHP*** (acrónimo de PHP: Hypertext Preprocessor)

De php podríamos decir

- Es un lenguaje de código abierto
- Muy popular (Podríamos pensar en un estándar?), una gran comunidad de soporte en internet que aporta, colabora y soluciona dudas
- especialmente adecuado para desarrollo web (Se puede usar como lenguaje de escritorio, pero no es su propósito).

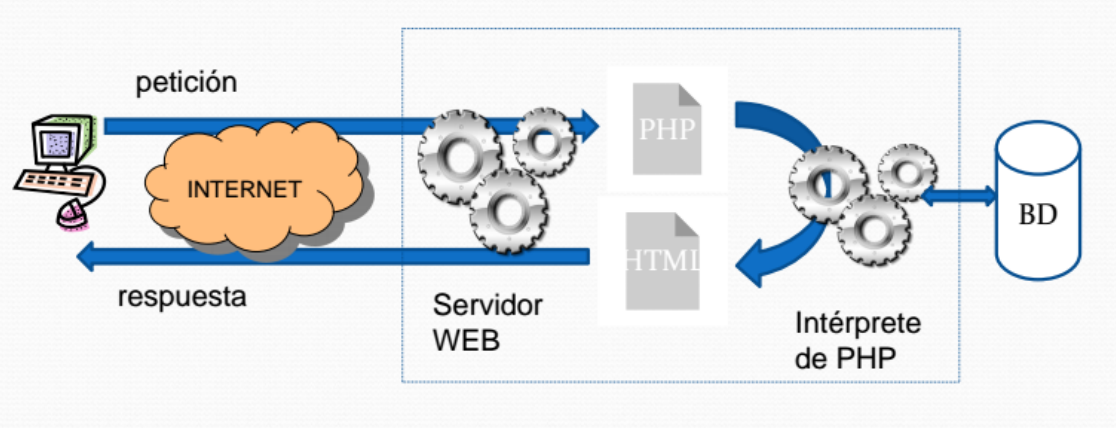
Qué es php

Tip:

- ***En las aplicaciones de desarrollo web***

1. Se ejecuta en el ***servidor web***
2. Es ***incrustado*** en HTML.
3. El cliente solo ve ***el resultado de la ejecución nunca el código***

Obtención del lenguaje de marcas para mostrar en el cliente



Puntos clave

El documento PHP, una vez interpretado correctamente en el servidor, produce una página HTML que será enviada al cliente.

El servidor en acción

Puntos clave

El código PHP está embebido en documentos HTML,
Esto permite introducir dinamismo fácilmente a un sitio web.

El servidor en acción

Puntos clave

El intérprete PHP ignora el texto del fichero HTML
Hasta que encuentra una etiqueta de inicio del bloque de código PHP embebido.

- Entonces interpreta las instrucciones hasta el final de etiqueta generando la salida correspondiente que se añade al documento html que se entrega al cliente (en caso de que las instrucciones lo generen)

Restricciones del servidor con php

- Como PHP se ejecuta del lado del servidor sólo puede tener acceso a los datos del propio servidor.
  - No puede acceder a los recursos del cliente
  - No puede saber qué hora es en el cliente
  - No puede acceder a los archivos del cliente
    - Salvo la excepción de las Cookies

Configuraciones

- PHP se puede instalar como un servicio independiente (PHP-FPM (FastCGI Process Manager)) o como un módulo de apache php5-mod. Realmente es más eficaz por temas de memoria que corra como un servicio independiente, siendo éste, un tema más de administración que de desarrollo.

Configuración

Por comodidad (todo centrado en el servicio de apache2) en este módulo lo hemos instalado como un módulo de apache, pero en producción se suele instalar como servicio independiente (en este caso se ha de rebotar el servicio de apache o ngix (según servidor) independientemente del servicio de php según los ficheros de configuración que se modifiquen en un momento dado. En cualquier caso, al instalar php, bien como módulo de apache o como servicio independiente, se crea un fichero de cofiguración dónde tenemos las diferentes directivas que podremos modificar (recordad xdebug que modificamos en php.ini).

Directivas de PHP.ini

```
http://www.php.net/manual/es/ini.list.php
```

Funciones que quedaron obsoletas en PHP 5.3.x

```
http://php.net/manual/es/migration53.deprecated.php
```

Características obsoletas en PHP 7.0.x

```
http://php.net/manual/es/migration70.deprecated.php
http://php.net/manual/es/migration70.incompatible.php
```

Cómo escribir PHP

- Dentro de páginas html

```
<?php
   instrucciones
?>
```

- Nosotros siempre usaremos este estilo para escribir código
- Lo podemos embeber en código html .... o no.

Otros modos menos usados

Estilo asp

```
<%
   instrucciones
%>
```

- Para ello hemos de tener habilitado la etiqueta de php.ini

```
asp_tags 1
```

Estilo corto

```
<?
   instrucciones
?>
```

- Para ello hemos de tener habilitado la etiqueta de php.ini

```
short_open_tag 1
```

Sintaxis para editores HTML

```
<SCRIPT LANGAGE="PHP">
   instrucciones
</SCRIPT>
```

- Guardamos el fichero con extensión .php
  - Así sabemos que el interprete php tiene que ejecutar código

Probando primer programa

Tip: existen una función llamada ***phpinfo()*** que vamos a probar la información que genera

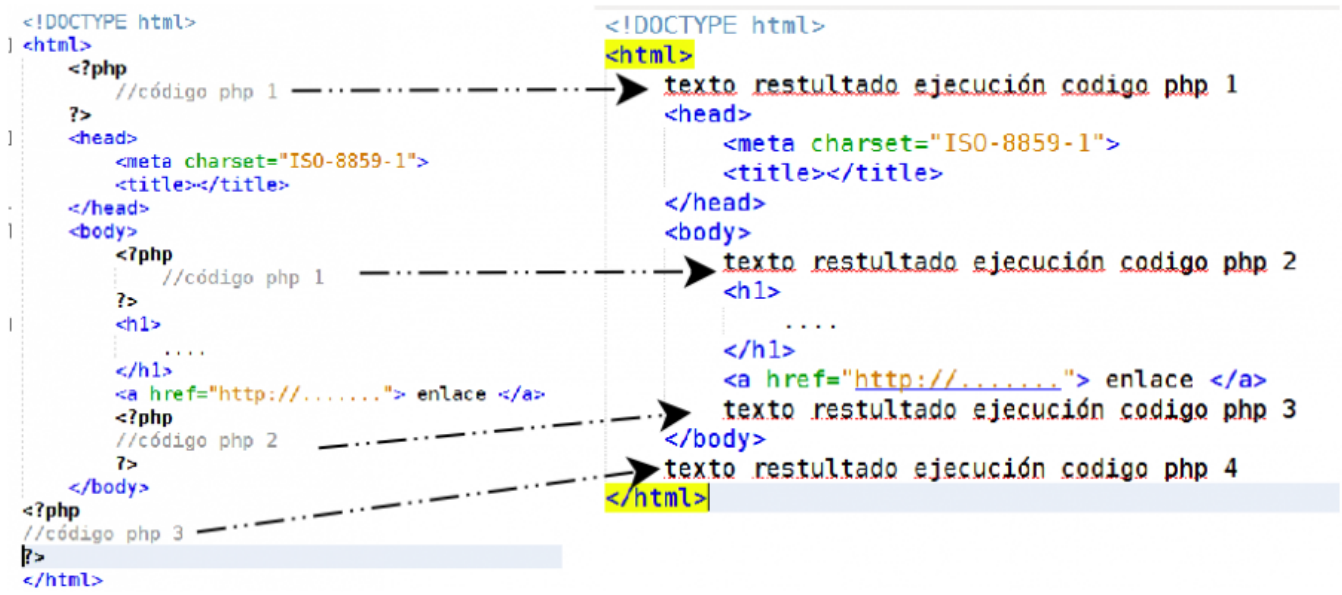
- Haz un programa que en php que ejecute la función ***phpinfo()***

Dónde poner el código embebido

Pregunta

Dónde escribir código php

- Donde queramos que se ejecute algo



- Se ejecuta como si fuera un solo programa

Escribir PHP con directivas de inclusión

- Podemos escribir el código php escribiéndolo en un fichero aparte y luego lo incluimos.
  - Incluimos el fichero explícitamente
  - Para ello usamos directivas o instrucciones del tipo ***include***

```
include('ruta/nobrefichero');
require('ruta/nobrefichero');
include_once('ruta/nombrefichero');
require_once('ruta/nombrefichero');
```

Ambas son palabras reservadas del lenguaje y sirven para incluir el contenido de un fichero con sentencias php en esa posición del código.

Include Vs Require

- Ambas incluyen el contenido de un fichero php en esa posición
  - Con include si no se encuentra se continúa ejecutando en esa posición
  - Con require si no está el fichero se detiene en ese punto la ejecución del script

include/require Vs include_once/require_once

- Ambas incluyen el contenido de un fichero php en esa posición
  - include/require siempre buscan e incluyen el fichero en esa posición
  - include_once/require_once antes de incluirlo mira a ver si ya lo incluyó previamente en cuyo caso ya no lo hace

Probando include require

Crea 4 ficheros con el siguiente código

fichero_include.php

```
<?php
  echo "<b><i>Hola desde un fichero include </b></i><hr />";
?>
```

fichero_include_once.php

```
<?php
 echo "<b><i>Hola desde un fichero include once </b></i><hr />";
?>
```

fichero_require.php

```
<?php
  echo "<b><i>Hola desde un fichero require </b></i><hr />";
?>
```

fichero_require_once.php

```
<?php
  echo "<b><i>Hola desde un fichero require once </b></i><hr />";
?>
```

Ahora crea un programa principal dónde uses las instrucciones de inclusión vistas en este apartado.

```
<?php

echo "<h2>Ahora vamos a incluir un fichero con include</h2>";
include 'ficheros/fichero_include.php';

echo "<h2>ahora vamos a incluir un fichero con require</h2>";
require 'ficheros/fichero_require.php';

echo "<h2>Ahora vamos a incluir un fichero con include_once</h2>";
include_once 'ficheros/fichero_include_once.php';

echo "<h2>Ahora vamos a incluir un fichero con require_once</h2>";
require_once 'ficheros/fichero_require_once.php';

echo "<h2>Ahora vamos a incluir un fichero que no existe  con include</h2>";
include 'ficheros/fichero_no_existe_include.php';
echo "Vemos que no pasa nada, por que el fichero no existe pero sigue <hr />";

echo "<h2>Ahora volvemos  a incluir un fichero con include_once</h2>";
include_once 'ficheros/fichero_include_once.php';
echo "Vemos que no pasa nada, por que el fichero ya se había incluido y no se vuelve a incluir<hr />";

echo "<h2>Ahora volvemos  a incluir un fichero con require_once</h2>";
require_once 'ficheros/fichero_require_once.php';
echo "Vemos que no pasa nada, por que el fichero ya se había incluido y no se vuelve a incluir<hr />";

echo "<h2>Ahora vamos a incluir un fichero con include para ver que sí que se vuelve a incluir</h2>";
include 'ficheros/fichero_include.php';

echo "<h2>Ahora vamos a incluir un fichero con require y vemos que sí se vuelve a incluir</h2>";
require 'ficheros/fichero_require.php';

echo "<h2>Ahora no incluimos con require un fichero que no existe</h2>";
require 'ficheros/fichero_no_existe_require.php';
echo "Esta línea ya no se imprimirá ni nada que vaya después de aquí";
?>
```

Tip:

- Siéntete cómoda en modificar el fichero y ver el resultado
- Observa que estas instrucciones no son exactamente funciones, por lo que no necesitan paréntesis (aunque se puede poner por mantener una homogeneidad con sintaxis del uso o invocación de funciones)

```
include 'ficheros/fichero_include.php'
//Es léxicamente y sintácticamente igual que
include ('ficheros/fichero_include.php');
```

Recursos de la Web

- php (http://php.net/manual/es/) página oficial en español LO MEJOR!!!!
- http://php.net/manual/es/
-

Comentarios

- Son ignorados por el intérprete, no generan instrucciones, pero se consideran parte del software
- En php tengo 4 formas de hacer comentarios

```
<?php
 /*
 Este código no hace nada pero muestra
 la sintaxis de los comentarios
 como este que ocupa varias líneas tipo lenguaje C o Java
 */
 $miVariable= 8;// Esta parte de la línea se ignorará
 $miVariable+= 8;# y esta parte de línea tambien
 echo "Valor de la variable $miVariable";
//Este es otro comentario de una sola línea
#Este es otro modo de hacer comentarios tipo script de linux
/**
Este comentario permite insertar información
Para que luego phpDocumentor genere una página web
Con la información de mi código
*/
```

- Este código nos visualizará lo siguiente

```
Valor de la variable 16
```

El resto del código será ignorado

- Aquí tienes la referencia para ver las diferentes directivas para generar código
- Más adelante en el curso las veremos.

Recursos de la Web

- Directivas para comentarios https://phpdoc.org/docs/latest/index.html

probando comentarios

- Escribe la siguiente función anterior en un fichero php

```
function miFuncion($num1, $num2){
    if ($num1>$num2)
        return $num1;
    else
        return $num2;
```

Ahora justo en la línea de encima de función escribe

```
/**
```

- y luego presiona intro
- Te debería de quedar

```
/**
 *
 * @param type $num1
 * @param type $num2
 * @return type
 */
function miFuncion($num1, $num2){
    if ($num1>$num2)
        return $num1;
    else
        return $num2;
```

Un programa: Un conjunto de instrucciones

Objetivo

Un programa es un conjunto de instrucciones

- Analizaremos las instrucciones que hay
- Posteriormente Veremos como se escriben en php

Planteando un lenguaje de programación

Léxicos

- Son las palabras reservadas del lenguaje

```
http://php.net/manual/es/reserved.keywords.php
```

Sintaxis

```
Reglas de construcción
Son las ya conocidas, pero veremos como se construyen las expresiones
```

Semántica

```
Habla del significado
```

Puntos clave

Estudiaremos alguna peculiaridad como el hecho de que php es un lenguaje *muy orientado a expresiones*

Instrucciones en un lenguaje de programación

1. Inicio Fin de bloque
2. Instrucción/función de leer del teclado, escribir por pantalla
3. Declaraciones (variables, constantes, funciones, clases, objetos, ...)
4. Asignación
5. Invocación (llamada a función o método)
6. Estructura de control (selectiva, iterativa)

Separando instrucciones

- Para separar una instrucción de otra usaremos ***;*** (punto y coma)
- Su uso es obligatorio a excepción de la última instrucción que se puede obviar
- Esto es por que el fin de código php ***?>*** implica esta instrucción
- Nosotros mejor lo usaremos siempre

Tip: Si solo queremos insertar una instrucción puede suele obviarse el ***;***

```
<html>
  ......
  <?php echo "hola" ?>
  <!-- instrucciones html -->
  <?php echo "otro hola" ?>
  <!-- mas instrucciones html -->
  <?php echo "otra instrucción " ?>
  .....
</html>
```

Tip: También puede haber ;

```
<html>
  ......
  <?php echo "hola" ;?>
  <!-- instrucciones html -->
  <?php echo "otro hola" ; ?>
  <!-- mas instrucciones html -->
  <?php echo "otra instrucción " ; ?>
  .....
</html>
```

Instrucción de inicio fin de bloque

```
{ //Instrucción de inicio de bloque
} //Instrucción de fin de bloque
```

Funciones de salida

- Construcciones básicas para salida de caracteres
- En PHP,en realidad no son funciones por lo que pueden ir sin paréntesis (con o sin paréntesis):

1. ***echo***
2. ***print***

- Existen otras funciones que iremos viendo según avance el curso

echo

- Es el uso más sencillo
- Imprime una cadena como argumentos
- En la versión ***sin paréntesis*** , también puedes pasar una lista de argumentos.

<?php

- echo "primer argumento", "segundo argumento", "tercer argumento"

?>

print

- Esta sentencia es igual en uso y funcionalidad que echo
- Tiene dos diferencias con echo

1. Sólo puede aceptar un argumento
2. Devuelve un valor booleano que representa si la sentencia ha tenido exito o no

Observa el siguiente código e indica si es o no correcto

1.
```
echo 'hola caracola', 'hola','como estás'
```
- ○ Correcto
- ○ Incorrecto

2.
```
print 'hola caracola', 'hola','como estás'
```
- ○ Correcto
- ○ Incorrecto

Declaraciones

Objetivo

Una declaración es una instrucción muy importante

- Aparentemente no hace nada (no tienen acción directa)
- Permite hacer luego cosas con los elementos declarados

Declaraciones

1. De variables
2. De constantes
3. De funciones
4. De clases
5. De objetos y recursos (clases ya creadas o incluidas)

Tip: Las funciones,clases y objetos los veremos en otro tema

Pregunta

- Qué es un ***tipo de dato***
- Qué es una ***variable***

Definición

Un tipo de dato es un conjunto de valores para los cuales hay definidos una serie de operaciones

Definición

Una variable es una posición de memoria que va almacenar algún valor de un determinado tipo, y cuyo contenido puede variar durante la ejecución de un programa

PHP

Tipado dinámico'

- Una características semántica muy, muy importante de php

Puntos clave

PHP es un lenguaje fuertemente tipado

- Los lenguajes de programación pueden ser mas o menos exigentes en cuanto a la declaración de los tipos de las variables para poder ser usados durante la ejecución de un programa.

tipado dinámico

- Lenguajes fuertemente tipos o débilmente tipados
- Esto tiene que ver con el hecho de que cada variable en un momento dado tiene un tipo, y lo podemos saber
- Esto ocurre en php

Tipado dinámico

- Php no es estricto en el tipo de dato de una variable en cuanto que éste puede cambiar durante su vida.

- En este sentido php es un lenguaje de ***tipado dinámico***, el tipo de la variable depende del valor que tiene en un momento dado o de los operadores que lo afecten.

Puntos clave

PHP es un lenguaje de tipado dinámico

Definir variables

- En php una variable es definida la primera vez que se usa.
- El tipo de la variable depende del valor que tenga asignado en un momento dado
- El identificador de php tiene que empezar por el signo ***$***
- En php las variables se representan con el signo ***$*** seguido de un carácter de subrayado o una letra y luego letras, números y caracteres de subrayado en cualquier orden y número.

PHP y variables

Resumen

1. Php es un lenguaje tipado
2. Php tipado dinámico
3. Php no es estricto en la declaración de variables (La declaración ocurre la primera vez que se usa)

identificador de variables

```
{{MRM_Puntos clave|identificador = $[_|a.zA..Z][_|a..zA..Z0..9]*
```

```
<?php
  $miVariable= 8;/*Variable de tipo entero*/
  edad = 5/*Error en el identificador*/
  $5edad = 5 /*Error en el identificador */
?>
```

Sensitive case?

- El lenguaje es sensible a mayúsculas y minúsculas
  - En los identificadores de variables ($edad != $Edad)
  - No lo es en nombres de funciones $calculaEdad(1990) != $CalculaEdad(1990)
  - No lo es en palabras reservadas (if o If o IF o iF,...)

Valores y tipos de datos

```
http://php.net/manual/es/language.types.intro.php
```

- En Php tenemos 8 tipos de datos

1. 5 tipos básicos o primitivos (un valor)
2. 3 tipos compuestos (conjunto de valores).

Tipos primitivos

1. ***boolean***: Valores TRUE y FALSE.
2. ***entero***: números enteros ... -2,-1,0,1,2 ...
3. ***real***: números reales
4. ***string***: Cadena de caracteres
5. ***NULL***: Valor null

Tipos compuestos

1. ***array***: conjunto enumerado por la clave de valores de diferente tipo
2. ***objeto***: Instancia de una clase en memoria
3. ***recurso***: objeto que permite utilizar elementos del sistema o **recursos**, (por ejemplo conector a una base de datos)

tipos básicos

1. **entero** *integer*

- Posible notación decimal/octal/hexadecimal

```
decimal  [0..9]+
hexadecimal 0x[0..f]+
octal 0[0..7]+
binario 0b[01]+
```

- Todos ellos pueden ser positivos o negativos

```
**$Numero=10;
*Octal
**$NumeroOctal=067;
*Hexadecimal
**$NumeroHex=0cA56B;
```

integer

- Al imprimirlos con print los veré con valor decimal
- Para verlos en otras base hay que usar printf o format o utilizar las conversiones dechex o dechex o octdec, ... que veremos en otro apartado.

cadena *string*

```
$frase="Esto es un literal de cadena de caracteres"
```

- - real o coma flotante ***float***

```
 $valor=$0.2345;
 $valor=.54;
 $valor=7E-12;
```

Booleano *boolean*

```
 $estado=TRUE;
 $estado=TrUe;
 $estado=falsE;
```

NULL

- un tipo especial que solo tiene ese valor
- Una variable tiene el valor null

1. Si aún no se le ha asignado valor, o éste se ha destruido (unset()
2. Si se le ha asignado explícitamente el valor NULL.

```
 $a=NULL;
 $a=null;
```

Tipos complejos

- Objetos básico en su aspecto de OOP
- Matrices o arrays muy muy utilizados
- Recursos este más que un tipo complejo es un tipo especial que hace referencia a un recurso externo referencia , como una conexión a una base de datos o como una referencia a un fichero pdf.
- Este tipo de variables las veremos más adelante

Ejercicio 1.- Declaración de variables

- Haz un programa donde declares variables de diferentes tipos
- Asigna los valores con diferente formato
- Visualiza sus valores

Constantes

Objetivo

Las constantes se declaran una vez
No se pueden modificar, solo usar

Constantes

- Se definen con la función ***define()***

```
define("IVA",0.21);
$total=$base*(1+IVA);
```

identificador

- Se usa el mismo criterio de construcción pero no empieza por $
- Se pueden definir y utilizar en cualquier momento que se necesiten.
- Para saber si una constante está definida ***defined()***

Constantes predefinidas

- Como en otros lenguajes, existen una serie de constantes predefinidas
- Nos las ofrece el entorno y dependerán de él para su valor
- PHP Ofrece un gran número de constantes predefinidas http://php.net/manual/es/reserved.constants.php
- En php hay 8 constantes que su valor puede cambiar dependiendo del entorno donde se ejecutan

```
constantes (http://php.net/manual/es/language.constants.predefined.php) predefinidas en php
```

Funciones

Objetivo

Las funciones es un elemento fundamental

- Permite crear código modular
- Una forma de estructurar nuestro programa

Declaración de funciones

```
function nombreFuncion ($paramFormal1, $paramFormal2 ,...){
  //Instrucciones de la función
  return $valorRetorno //Opcionalmente en caso de que devuelva algún valor la función
}
```

- Es importante diferenciar entre declarar una función e invocar a una función
- Algo obvio, pero importante
- En la declaración tenemos tres partes

1. nombre o identificación de funciones
2. parámetros formales entre paréntesis (Estos han de existir, aunque no haya parámetros)
3. Cuerpo de la función, dentro de él puede estar la instrucción return, en cuyo momento termina la ejecución de la función y se vuelve a la siguiente instrucción del programa, siguiente a la invocación de la función.

Identificador de función

- El nombre de función es un identificador que empieza por una letra o guión bajo, seguido 0 o muchas letras, números o guiones bajos

Tip: Expresión regular para el identificador de funciones

```
[a-zA-Z_f_][a-zA-Z0-9_]*
```

Parámetros formales

- Son nombres de variables que usaré al escribir el código o cuerpo de la función
- El nombre ha de ser significativo y se convertirán en variables locales a la función
- Una vez que se termina la función estas variables desaparecerán de memoria

parámetros formales

Los parámetros formales son variables locales a la función

Ejercicio usando funciones

Haz un programa donde en el programa principal se creen dos variables $a y $b

- Crea una función que reciba como parámetros locales ***$a*** y ***$b***
- La función visualizará el valor de las variables, las modificará y las volverá a visualizar
- El programa principal

1. asignará valor a las variables
2. las visualizará
3. invocará a la función
4. volverá a visualizar las variables

- Una posible solución

```php
<?php
    function a($a, $b){
        echo "Dentro de la función visualizando valores <hr />";
        echo "Valor de los parámetros \$a = $a \$b = $b <br />";
        $a+=5;
        $b+=5;
        echo "Valor de los parámetros \$a = $a \$b = $b <br />";
        echo "Salgo de la función";
    }
    //Ahora considero programa principal
    $a=100;
    $b=200;
    echo "En el main antes de invocar a la función visualizando variables<hr />";
    echo "Valor de variables \$a = $a \$b = $b <br />";
    a($a,$b);
    echo "En el mail  después de invocar a la función visualizando variables<hr />";
    echo "Valor de variables \$a = $a \$b = $b <br />";
?>
```

Parámetros formales: Valores y referencias

- Cómo hemos visto, los parámetros formales son valores pasados en la invocación a la función
- Si queremos que la función pueda modificar el valor de los valores de los parámetros, en este caso hemos de pasarlos por referencia
- En este caso lo que ocurre en realidad es que pasamos la dirección de memoria dónde se guarda el valor.
- La dirección de memoria, no la podremos visualizar ni operar con ella, pues en php no existe la aritmética de punteros o direcciones de memoria

Parámetros formales

Valores y referencias

Para pasar el parámetro por referencia, simplemente hay que poner el símbolo de dirección de memoria **&** antes del nombre de la variable en la declaración de parámetros

```php
function nombre_funcion(&$paramRef1, &$paramRef2, $paramVal1){
   ...
}
```

Ejercicio usando funciones parámetros

Haz un programa donde en el programa principal se creen dos variables $a y $b y $c

- Crea una función que reciba como parámetros locales ***&$num1***,***&$num2*** y ***$num3***
- La función visualizará el valor de las variables, las modificará y las volverá a visualizar
- El programa principal

1. asignará valor a las variables
2. las visualizará
3. invocará a la función
4. volverá a visualizar las variables

```php
<?php
    function a(&$num1, &$num2, $num3){
        echo "Dentro de la función visibilizando valores <hr />";
        echo "Valor de los parámetros \$num1 = $num1 \$num2 = $num2 \$num3 = $num3<br />";
        $num1+=5;
        $num2+=5;
        $num3+=5;

        echo "Valor de los parámetros \$num1 = $num1 \$num2 = $num2 \$num3 = $num3<br />";
        echo "Salgo de la función";
    }
    //Ahora considero programa principal
    $a=100;
    $b=200;
    $c=300;
    echo "En el main antes de invocar a la función visualizando variables<hr />";
    echo "Valor de variables \$a = $a \$b = $b \$c = $c <br />";
    a($a,$b,$c);
    echo "En el mail  después de invocar a la función visualizando variables<hr />";
    echo "Valor de variables \$a = $a \$b = $b \$c = $c <br />";
?>
```

Invocando funciones

- Una vez creada una función la podemos invocar como si fuera una instrucción del lenguaje
- No sin razón en determinados ambientes se conoce a las funciones y procedimientos como instrucciones virtuales ...
- En php puedo invocar a una función antes de declararla, siempre que la declare en el mismo fichero

ejemplo invocación a funciones

Tip: Este código funcionará correctamente

```
<?php
  a(5,6);
  /*Mas instrucciones*/
  function a ($a, $b){
      echo "valor de $a";
      echo "valor de $b";
  }
```

ejemplo invocación a funciones

Tip: Este código no funcionará

```
<?php
      a(5,6);
      /*Mas instrucciones*/
      include ("funciones.php");
?>
```

- Contenido del ficheor funciones.php

```
<?php
function a ($a, $b){
          echo "valor de $a";
          echo "valor de $b";
}
?>
```

Funciones de php de tipos y valores

```
http://php.net/manual/es/ref.var.php
```

- Existen una serie (muchas) de funciones que son interesantes de conocer
- Estas funciones ya están creadas y se pueden usar directamente
- Están relacionadas con los tipos de datos y valores
- Alguna de ellas son extremadamente útiles y utilizadas, por ejemplo antes de procesar un dato, hay que ver que dicho dato tenga valor.
- A continuación trataremos alguna de ellas

var_dump (http://es1.php.net/manual/es/function.var-dump.php)

```
void var_dump($expresion)
```

- Nos da información sobre la estructura de un valor resultado de una expresion

isset (http://es1.php.net/manual/es/function.isset.php)

```
bool isset ( $variable )
```

- verifica que una variable tiene valor (está definida y no tiene un valor null)

```
<?php
 $VariableValor= 5;
 print ("El valor de la variable es $VariableValor");
 print ("El valor de otra variable es $OtraVariableValor");
 if (isset($VariableValor))
     print ("VariableValor tiene valor asignado");
 else
     print ("VariableValor no no tiene valor asignado");
 if (isset($OtraVariableValor))
     print ("OtraVariableValor tiene valor asignado");
 else
     print ("OtraVariableValor no no tiene valor asignado");
  ?>
```

empty (http://es1.php.net/manual/es/function.empty.php)

```
bool empty ($varriable)
```

- Determina si una variable no existe. Devuelve true si no existe o su valor está vacío

Actividad

Probamos las fuciones var_dump() que nos da información sobre el valor y el tipo

Actividad

Usando la función xxxyyy donde xxx e yyy será dec oct bin o hex para convertir el valor de un sistema numérico a otro

Actividad

- Define las siguientes variables que se especifican en el código siguiente y verifica el resultado con empty()

```
$num=0;
$nombre="";
$nombre=null;
$nombre="0";
$pregunta = FALSE;
```

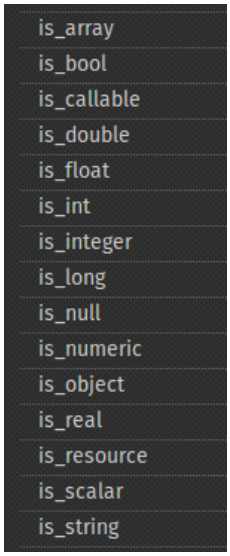
gettype (http://es1.php.net/manual/es/function.gettype.php)]

- Devuelve el tipo de una variable

```
string gettype($variable)
```

[1] (http://es1.php.net/manual/es/function.is-bool.phpis_bool)is-double (http://es1.php.net/manual/es/function.is-double.php) is-int (http://es1.php.net/manual/es/function.is-int.php), is-xxx

- son funciones donde xxx especificado en el último nombre, puede ser cualquiera de los tipos



- Todas ellas devuelve un booleano que indica si la variable, valor o expresion es o no de ese tipo,

```
string is_int($variable);
string is_double($variable);
string is_bool($variable);
string is_integer($variable);
string is_null($variable);
string is_string($variable);
...
```

Actividad

Visualizar de qué tipo es la expresión mostrada en el código siguiente y visualiza el valor de la expresión

```
$a=5;
```

unset (http://php.net/manual/es/function.unset.php)

- Destruye la variable especificada perdiéndose su valor

void unset ($var)

Cadenas

Objetivo

Cómo se trabaja con estos valores
como puedo concatenar y

- Cómo incluir en una cadena
  - valores de variables
  - retorno de funciones
  - valores de expresiones

Php y los valores de tipo cadena

- En php las cadenas de caracteres, son expresiones literales
- Tenemos 4 maneras diferentes de poder expresar una cadena de caracteres como un literal
- Comillas dobles ""
- Comillas sencillas ''
- Sintaxis ***heredoc***
- Sintaxis ***nowdoc***

Comillas dobles

- En ellas se interpretan los caracteres especiales

1. **$** seguido de un nombre, interpreta que es una variable y toma su valor (null si no tiene valor o no está definida
2. **** es un carácter de secuencia de escape, e interpreta que el carácter siguiente tiene un significado especial \\ \a \n \r \t , ...

- Si queremos que se ignore un carácter especial, éste ha de ir precedido por el caracter ****

```
$nombre = 'pedro';
echo "El valor de la variable \$nombre es $nombre";
```

- El resultado sería

```
El valor de la variable $nombre es pedro
```

Comillas simples

- En ellas solo se interpreta el carácter, seguido de \ o bien seguido de la barra invertida **** comilla simple **\'**
- El resto de caracteres no se interpretan

```
$nombre = 'pedro';
echo 'El valor de la variable $nombre es $nombre y \\ \'texto\' si que se ve entre comillas simples';
```

- La salida sería

```
El valor de la variable $nombre es $nombre y \ 'texto' si que se ve entre comillas simples';
```

Heredoc

- Este tipo de expresión de string es útil para especificar cadenas largas en multilíneas
- Se comporta como un string entre comillas dobles para el tema de interpretar y escapar ciertos caracterres
- Se establece con el operador **<<<**
- A continuación viene un identificador
- Después empieza a especificarse la cadena de caracteres
- Para finalizarla se escribe en una nueva línea el identificador

```
<?php
$frase = <<<FINAL
Esta es una cadena
de caracteres que se asignará
a la variable frase
y termina con la palabra
con la que hemos empezado
FINAL;
<?
```

- La palabra final no debe tener ningún espacio ni tabulador antes

NewDoc

- Es igual que heredoc , pero sin interpretar los caracteres especiales salvo **\\ \.**
- O sea que es como un entrecomillado sencillo
- La sintaxis es igual que la de heredoc, pero a diferencia el delimitador que se especifica al principio debe de ir entrecomillado con comillas simples

```
<?php
$nombre=pedro;
$frase = <<<'FINAL'
El valor de $nombre
es $nombre, pero aquí
no lo veo por que es newdoc
FINAL;
```

Estructuras de control

Objetivo

Determinan el flujo de ejecución de un programa

- Tenemos tres estructuras de control

- Veremos cómo se implementan en PHP

Estructuras de control 1

A continuación veremos las estructuras de control Son de tres tipos

1. Selección
2. Iteracción
3. Secuenciales

- Para construirlas necesitamos operadores

Selección if

- Sentencia que evlúa una expresión booleana y ejecuta o no en función de que dicha expresión sea true o false

```
if (condicion)
    Sentencia 1;
```

```
if  (condicion){
    Sentencia_1;
    Sentencia_2;
}
```

Sentencias de control

if (expresion)

```
   Sentencia_1;
```

else

```
    Sentencia_2;
```

</source>

- También existe la opción elseif donde aportaremos una condición que se ha de cumplir para que se ejecuten las sentencias que a continuación acompañan.

Estructura de control

```
if (expresion){
   sentencias;
}
elseif (expresion){
   sentencias;
}
else{
   sentencias;
}
```

- Alternativamente puede usarse esta sintaxis que es usada cuando se quiere intercalar condigo html fuera del php.
- También se puede usar la sintaxis vista anteriormente, pero parece que esta quede más compacta.

```
if (condicion):
   Sentencia 1;
endif;
```

Ahora lo vemos con código html

```
<?php if (true): ?>

    <h1>Esta frase seguro que aparece ahor</h1>
    <!--escribirmos código html -->
<?php else: ?>
    <h1>Aquí escribiré poco ya que no va a aparecer nada</h1>
    <!--escribimos código html -->
<?php endif ?>
```

Operadores ternario

- Es una forma más compacta de un if else con una única instrucción.

```
Expresión? SentenciaOKExpresion : SentenciaNoOkExpresion
```

Actividad

Programa que me de si un número aleatorio es par o impar

Seleccion switch

- Este es un selector múltiple
- La sentencia case puede albergar cualquier valor de un tipo simple, no está limitado a enteros como en otros lenguajes
- Estructura indicada cuanto tengamos más de dos casos ante una variable o situación que evaluemos excluyentes entre sí

Switch

```php
<?php
switch ($nombre){
    case "Maria":
        echo "eres una chica";
        break;
    case "Pedro";
        echo "eres una chico";
        break;
    default:
        echo "no se qué nombre tienes";
}
?>
```

Iteración while

- Como en todos los bucles debemos siempre tener encuenta

1. inicializar la variable de control
2. actualizarla correctamente dentro del bucle
3. realizar de forma correcta la evaluación de condición (< o <=), (> o >=), ...

```php
<?php
$i = 1;
while ($i <= 10) {
    echo "iteracción número ."$i++;
}
?>
```

- Alternativamente podemos usar la siguiente sintaxis

```php
$i = 1;
while ($i <= 10):
    $i++;
    echo "iteracción número ."$i;
endwhile;
?>
```

Iteracción do-while

- Este tipo de bucle donde seguro que al menos se ejecuta un iteracción
- Respecto al anterior nos ahorra una comparación.

```php
<?php
$num=10;
$resultado=1;
*Esta es la única sintaxis posible con este tipo de sentencia
do {
   $resultado:=$resultado*$num;
   $num--;
} while ($num>0);
?>
```

Iteración for

- Es un bucle de tipo contador

```
for (expresion_inicial; condicion;expresion_actualizar){
      sentencias;
}
```

Estructura for

- tiene tres partes

expresion_inicial

Se ejecuta una sola vez al comienzo del bucle. se usa para inicializar variables

condición

Es una expresión booleana que se evalúa en cada interacción
Si da un valor false, ya no se ejecuta ninguna vez
Si no hay expresión se toma como true
En este caso para que el bucle no sea infinito deberá llevar algún break (instrucción de terminación de bloque) en algún momento

Estructura for

condición

```
<?php
for ($a=0; ;$a++){
    echo "$a*$a=".$a*$a."<br>";
    if ($a==10)
        break;
}
?>
```

Estructura for

expresion_actualizar
Esta expresión actualiza el valor de alguna/as variables
Se ejecuta en cada interactivo

- El ejemplo anterior

```
<?php
for ($a=0;$a<10 ;$a++){
    echo "$a*$a=".$a*$a."<br>";
}
?>
```

Operadores y expresiones

Objetivo

Son partes de las frases de un lenguaje de programación

Operadores

- Son símbolos que realizan acciones sobre operandos y dan como resultado un valor
- Tenemos diferentes tipos de operadores en función del tipo de operandos y del resultado

operadores aritméticos (+,-,*,/,%,, ++, --)**

- Retorna un valor numérico
- el ++, -- son valores de autoincremeto y autodecremento, pueder ser pre o post

Operadores

```
$a=5;
if ($a++==5)
    echo '$a que vale '.$a.' dice que vale 5 ???? <br>'
        . 'Esto es por que primero compara y luego incrementa<br>';

echo 'ahora $a vale '.$a.'<br>';

if (++$a==6)
    echo 'esto nunca saldrá ya que $a se incrementa antes de comparar';
else
```

```
    echo 'efectivamente ahora $a ya no vale 6 sino '.$a.'<br>';
?>
```

Operadores

- El código anterior genera la siguiente salida

```
 $a que vale 6 dice que vale 5 ????
 Esto es por que primero compara y luego incrementa
 ahora $a vale 6
 efectivamente ahora $a ya no vale 6 sino 7
```

operadores comparación (==,<,>,>=,<=,<>,!=,===,!==)

Este tipo de operadores genera un booleano como resultado de evaluar la expresión

Puntos clave

{{{1}}}

Operador == Vs ===

```
$num=1;
if ($num==true)
    echo '$num es igual a true<br>';
if ($num===true){
    echo "esto nunca se ejecutará";
}else
    echo '$num no es exactamente igual a true';
```

Operador == vs ===

- El código anterior generaría la siguiente salida

```
$num es igual a true
$num no es exactamente igual a true
```

- Ver la sección ***comparación de tipos*** de la página oficial

```
http://php.net/manual/es/language.operators.comparison.php
```

operadores de concatenacón(.) concatena cadena de caracteres.

El operador + no está sobre cargado, observa el siguiente código

```
$nombre="Maria";
$apellido = " de la Oh";

$nombreCompleto = $nombre.$apellido;
echo "el valor de nombre completo es $nombreCompleto ---<br>";

$nombreCompleto = $nombre+$apellido;
echo "el valor de nombre completo es $nombreCompleto --<br>";
```

La salida del código anterior sería

```
el valor de nombre completo es Maria de la Oh ---
el valor de nombre completo es 0 --
```

Operadores de asignación (= , =>)

Se pueden combinar con los aritméticos (+=, *=, ...) y con los de concatenación (.=)

En este caso el valor de la variable de la izquierda se toma como primero operando

Operador de asignación

```
<?php
$b=1;
for ($a=0;$a<10;$a++){
    $b*=10;
    echo 'valor de $b ='.$b.'<br>';
}
?>
```

- El código anterior genera la siguiente salida

```
valor de $b =10
valor de $b =100
```

```
valor de $b =1000
valor de $b =10000
valor de $b =100000
valor de $b =1000000
valor de $b =10000000
valor de $b =100000000
valor de $b =1000000000
valor de $b =10000000000
```

operadores de ejecución (``)

PHP proporciona un operador especial que permite ejecutar sentencias

- Observa el siguiente código

```
<?php
$Discos = `df`;
echo "<pre>$Discos</pre>";
?>
```

- El código anterior generará la siguiente salida

```
Filesystem     1K-blocks     Used Available Use% Mounted on
/dev/sda5       86378608  6072360  75895384   8% /
none                   4        0         4   0% /sys/fs/cgroup
udev             4023720        4   4023716   1% /dev
tmpfs             806904     1384    805520   1% /run
none                5120        0      5120   0% /run/lock
none             4034504     6588   4027916   1% /run/shm
none              102400       28    102372   1% /run/user
/dev/sda7      101797224 40480360  56122728  42% /home
```

Invocando funciones del sistema

- El operador anterior (comillas invertidas) es igual que la función ***shell_exec()***

```
http://php.net/manual/es/function.shell-exec.php
```

operadores lógicos (and,&&, or, ||, xor !)

```
http://php.net/manual/es/language.operators.logical.php
```

Funcionan por cortocircuito
El operador ***xor*** da verdad si los operando son de diferente valor uno true y el otro false
La notación ***and*** y ***&&*** representan el mismo operador, igual ocurre con ***or*** y **||**

- La diferencia entre los operadores es la prioridad

```
http://php.net/manual/es/language.operators.precedence.php
```

Uso de formularios para leer datos del cliente

De WikiEducator
< Usuario:ManuelRomero

Formularios: Pasando información del cliente al servidor

¡Los formularios como entrada de datos a nuestros script

PHP Un lenguaje de script al lado del servidor

Formularios | Ejercicios | Práctica | Volver

INDICE

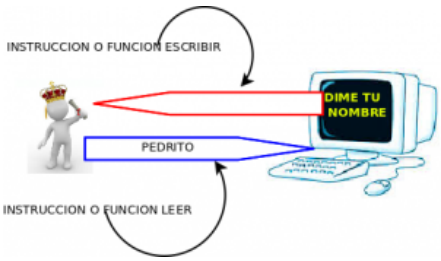
Contenido

Show presentation

Introducción

- Todos los lenguajes de programación tienen primitivas o incluso instrucciones propias
- Ya vimos que un programa necesita interactuar con el usuario

1. Leer valores del teclado
2. Mostrar resultados en pantalla



- En el caso de PHP, hemos visto alguna primitiva para mostrar valores por pantalla (En realidad lo que hace es escribirlas al fichero html que entrega al cliente), como son ***echo*** y ***print***
- Ambos dos son son instrucciones del lenguaje, ver la diferencia

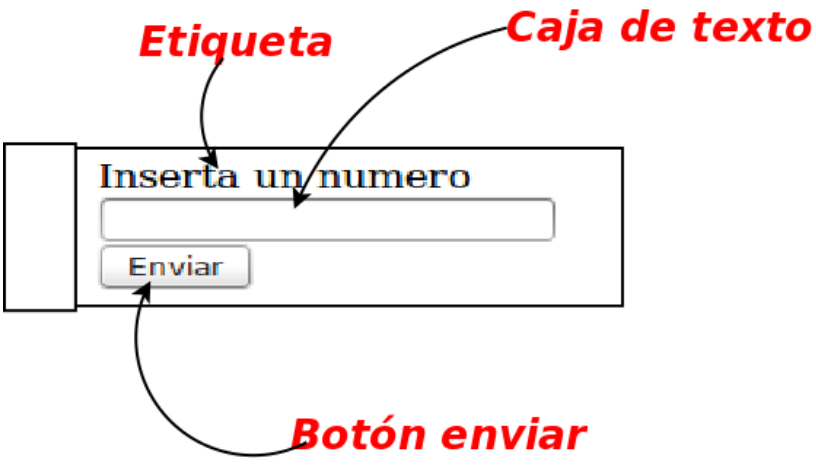
Actividad

Completa el siguiente programa

```php
<?php
$n1=1;
$n2=2;

//Usando echo con múltiples parámetros
//Visualiza la suma, la resta, y la multiplicación
//Al ser varios parámetros usa las comas para separar uno de otro
echo "Usando echo <br/>";

//???????
//Usando print, solo puedo usar un parámetro, así que tenemos que concatenar (operador .)
//Recupera el valor que retorna print y visualizalo
echo "<br/> usuando print<br/>";

????????
?>
```

Cómo leer datos de usuario

- Nos falta ver como podemos hacer que el cliente (a través del navegador) aporte valores al programa escribiéndolos por el teclado
- Para leer datos lo hacemos mediante un formulario (hay más métodos, como leer un fichero ,bases de datos, ...)
- El formulario será parte de la página del cliente.



Leyendo del usuario

- En él tendremos cajas de texto donde en el navegador el usuario podrá escribir contenido
- En los diferentes elementos de entrada de un formulario , como una texto, el usuario podrá escribir valores.
- Al darle el botón enviar dichos valores irán al servidor, en seguida veremos como leerlo en el servidor
- Repasemos como crear formularios en el cliente y lo que más nos interesa, como leerlos en el servidor

Creando un formulario

- Esta parte la veis el módulo de diseño de interfaces , no obstante comentaremos lo que aquí vamos a utilizar
- Un formulario se establece con la etiqueta ***form***

Etiqueta form

Definición

Para la programación servidor, entendemos por formulario una sección del código html que va a poder contener, además de otros elementos varios objetos gráficos con los que el usuario va a poder interactuar e insertar valores para que éstos lleguen al servidor

Atributos de la etiqueta

- Etiqueta form con una serie de atributos, de los que nos interesan dos principalmente

1. action especifica el fichero que se invocará al servidor. Este fichero contendrá el código php que queremos que se ejecute
2. method especifica el modo en el que se van a pasar los parámetros (valores introducidos a los diferentes objetos del formulario, o que tengan asignados por defecto).

GET o POST

```
<form action="mifichero.php" method="POST"
 .......
</form>
```

- Por defecto los valores son pasados por GET
- Este método es fácil de ver pues se viauliza en el URL, apareciendo como parte de él separado por el signo interrogación con parejas Variable valor

Ejemplo

{{{1}}}

Atributos

- En este caso estamos indicando que cuando se envíe el formulario, se intentará ejecutar un fichero llamado ***mifichero.php***. La ubicación del fichero como no se especifica se supone en el mismo sitio donde estaba el fichero que actualmente está viendo el cliente.
- También se especifica que los valores enviados con el formulario irán en el cuerpo de documento usando el protocolo http, y no en el URI con el signo ? cono ocurre si se especificara GET

Puntos clave

Tan inseguro es usar GET como POST. si queremos garantizar seguridad debemos usar https y no http

- Por supuesto hay más atributos, el id es importante para poder acceder a ese elemento con javascript; estos son los que nosotros debemos conocer para usar
- También es interesante el atributo0 ***enctype'*** *que permite usar algún tipo de cifrado para enmascarar la información que se envía, pero insisto en usar https si se quiere confidencialidad con un nivel aceptable de seguridad.*
- Este atrubuto es importante cuando en lugar de input de texto enviemos ficheros u otros contenidos diferentes.

Elementos dentro del formulario

- Dentro del formulario debemos poder recoger información que el cliente nos facilite
- Al menos deberíamos de conocer dos elementos ***input*** y ***button*** o bien ***submit***
- El input representa una caja de texto
- El submit es un botón que tiene automatizada la acción de enviar el formulario al hacer click sobre él.

Creando formularios

Es interesante ojear esta sencilla página que te informa de como hacer formularios

```
http://www.aulaclic.es/html/t_8_1.htm
```

- Elemento input, en él especificaremos los siguientes atributos

type

indicaremos el tipo de elemento de entrada (text, password, email, checkbox...)
Aquí podemos ver una lista de posibles valores, tened en cuenta que con html5 se introdujeron 13 nuevos tipos.

```
http://www.w3schools.com/tags/att_input_type.asp
...)
```

Atributos

También es interesante el tipo hidden (especialmente usado para pasar valores del cliente al servidor de forma transparente para el usuario.

name

indicaremos el nombre asociado a este input.
Con este identificador podremos en el servidor recuperar la información.

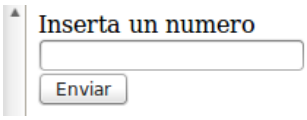
value

Es el valor que tiene el input. Si queremos que por defecto tenga un valor

- Dentro del form necesitaremos al menos un input y un submit
- EJ. en el emisor tenemos

```
<!DOCTYPE html>
<html>
    <head>
        <title>Tabla de multiplacar</title>
    </head>
    <body>
        <form action="tabla.php" method="GET">
            Inserta un numero <br>
            <input type="text" name="numero"/>
            <br/>
            <input TYPE="submit" VALUE="Enviar"/>
        </form>
    </body>
</html>
```

- Y obtenemos la siguiente imagen



1. Al presionar el botón de enviar se envía la página al servidor
2. La página que especificamos en el botón *action* la gestiona *tabla.php* en este caso
3. En el servidor para recuperar el valor utilizaremos la variable *supergobal*
4. una tabla es una estructura indexada por índices
5. leeremos el índice nombre de variable de la tabla
6. $_GET o $_POST dependiendo de el método de envío

```
Valor numérico introducido: <?php echo $_GET['numero'] ?>
```

Obtener datos de un formulario

- Una vez que estamos en el servidor, los datos son pasados del cliente al servidor usando las variables superglobales o matrices $_POST $_GET, $REQUEST
- Dependerá del modo en el que pasemos los datos del formularios desde el cliente

```
atributo method del form
```

- Para leer el datos indexaremos la matriz por el valor del atributo ***name*** de ***input*** correspondiente
- Por ejemplo en el cliente tenemos

```
....
<form method=POST action ="resuelve.php">
....
 <input type=text name=nombre>
...
</form>
```

- En el servidor el servidor el fichero ***resuelve.php***

```
....
$nombre = $_POST['nombre'];
....
//También podríamos $_REQUEST['nombre'];
```

Verificando si una variable existe (aunque tenga valor null).

- Es especialmente importante en muchas ocasiones ver si una variable tiene o no valor
- No sabemos si el usuario a insertado o no valor en el campo de texto
- Para ello usaremos la función ya conocida ***isset($variable)***, donde $variable es la variable que queremos ver si tiene valor

</source>

- A continuación vamos a realizar una serie de prácticas con formularios

Actividad

Haz un formulario en el que insertemos un número y el servidor web nos visualice la tabla de multiplicar

- Comprobaremos previamente que la variable exista y tenga un valor numérico

1. ***isset($variable)*** , Para ver que exista la variable
2. ***is_null($variable)*** me dice si la variable es nula ojo puede tener el valor null y no ser nula
3. ***is_numeric($variable)*** me dice si el valor de la variable es numércio

```
$nombre = ""; //nombre tendrá el valor nulo pero es de tipo null
if ($nombre==null) //Me dará verdad
   ......
if (is_null($nombre)) //Me dará falso
```

- A continuación vamos a ver como usar y leer datos de un formulario.

Formulario

Realiza un formulario donde pidamos al usuario datos para confeccionar una ficha

- Nombre
- Apellidos
- Dirección
- Fecha de nacimiento
- Edad
- Idiomas que habla de entre 4 idiomas (Checkbox)
- Si es hombre, mujer o no quiere informar de ello (radio)
- Dirección de correo electrónico.
- Estudios realizados entre ESO, BACHILLER, CICLO FORMATIVO, GRADO UNIVERSITARIO (select)

Filtrando valores

- Independientemente de que el se validen/verifiquen valores en el cliente, conviente verificarlo siempre en el servidor
- Para ello podemos usar la función filter http://php.net/manual/es/function.filter-var.php.

Tenemos la opción de filtar_var() y filter_input().

filter_var($variable, $filtro)

1. $variable . Es la variable a filtar
2. $filtro. Es el tipo de filtro que se quiere aplicar. Para ver los tipos de filtros, consultamos a la página web http://php.net/manual/es/filter.filters.validate.php

filter_input($tipo_entrada. $variable, $filtro)

1. $tipo_entrada: Uno de los siguientes: INPUT_GET, INPUT_POST, INPUT_COOKIE, INPUT_SERVER o INPUT_ENV.
2. $variable: como en el caso anterior
3. $filtro: como en el caso anterior

- Ambas funciones retornan el valor de la variable requerida, o false si el filtro falla o null, si la variable no tenía valor.

Actividad

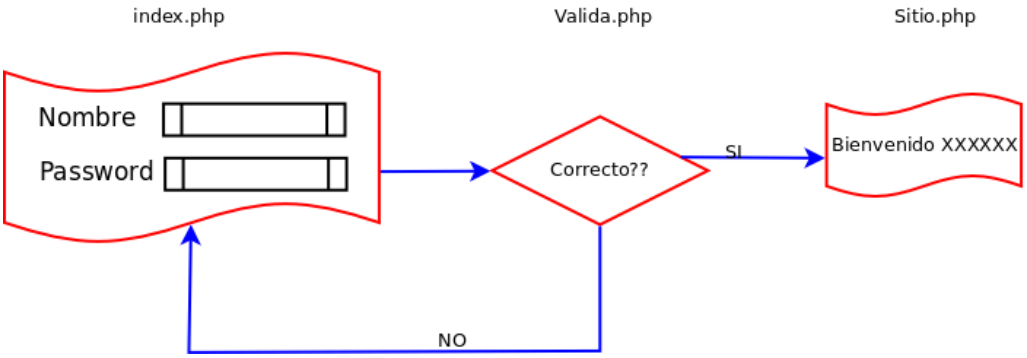
Filtra los valores en un formulario y verifica el tipo de la variable introducido

- Hagamos los siguientes ejercicios

http://www.tecn.upf.es/~ocelma/cpom/practicas/

Redirigiendo páginas

- Imaginemos que queremos hacer una página donde pidamos al usuario nombre y password
- El password va a ser 12345. Si el password es correcto iremos a otra página en la que le queremos dar la bienvenida con el nombre que introdujo
- Pensemos en como podemos pasar ese nombre a la página



- Analicemos las maneras de hacerlo, pero previamente veamos una muy interesante función
- La usaremos mucho mucho, y sirve para invocar a otras páginas en un momento dado

header(....);

header() se usa enviar encabezados HTTP sin formato. En esas cabeceras es cuando invocamos a una determinada url que queremos cargar, así que es ahí donde podemos hacer referencia a la página que queremos ver.

- Es muy importante saber que ***header()*** debe ser llamado antes de mostrar nada por pantalla
- Aquí se puede acceder a la referencia oficial.

```
http://es.php.net/manual/es/function.header.php
```

- Por ahora la usaremos de dos maneras para un mismo cometido

Cargar una página inmediatamente

```
header("Location:URL_de_la_página");
```

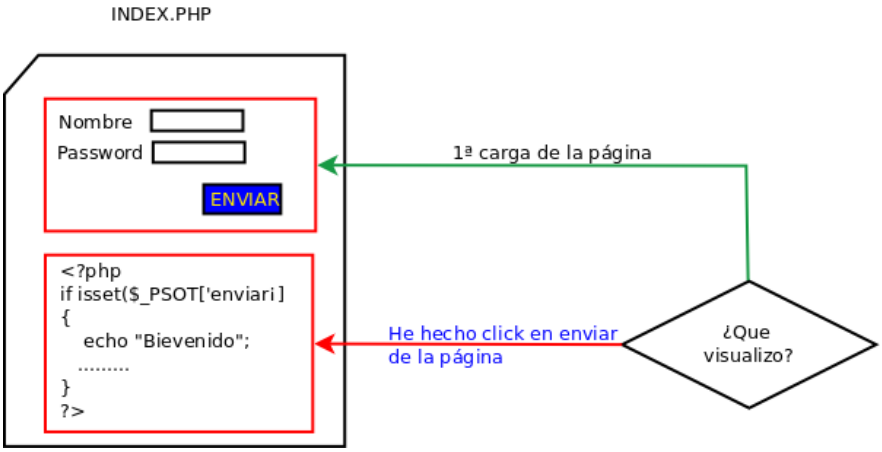
Cargar una página con un tiempo de demora (por ejemplo para leer un mensaje)

```
header ("Refresh:5; url=URL_de_la_pagina");
```

- Ahora estamos en condiciones de probarlas con el ejemplo anterior.
- Es muy importante entender las acciones que se van haciendo

Referenciando la propia página

- A veces puede ser que en la propia página tengamos el código que queremos que se ejecute cuando hacemos un click en el botón submit.
- Esto simplifica el número de páginas que tenemos en nuestro desarrollo
- En este caso tenemos la siguiente situación



- En la imagen vemos una forma de proceder
- Creamos una página web
- Dentro de la página tenemos que ver si es la primera vez que se carga la página o no
- Otra forma de verlo es si se ha cargado la página porque hemos puesto el url en el navegador, o porque hemos hecho click en el botón submit del formulario
- Dentro del código esto lo podemos saber interrogando si existe la variable ***$_POST['enviar']***

Actividad

Haz una página de bienvenida que muestre los datos de usuario y pass al acceder al sistema

- Se tiene que hacer en una única página

Pasando información de una página a otra

- La programación web utiliza el protocolo http para la transferencia de los datos
- http es un protocolo sin estado

Puntos clave

1. La programación web está basada en el protocolo ***http***
2. El protocolo ***http*** es un protocolo sin estado

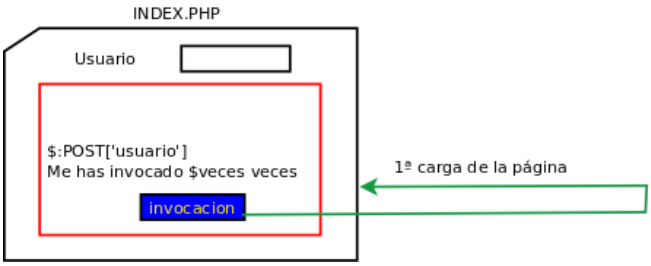
- Cada vez que cargamos una página o hay una solicitud el servidor web entrega al cliente la página ejecutada
- En caso de que la página ejecutada tenga código php (extensión del fichero), el servidor web ejecuta dicho código y entrega como parte de la página el resultado de la ejecución
- El servidor no tiene en cuenta a quién entrega la página, no sabe si es la primera vez que te entrega la página o la enésima vez que lo hace
- Si quiere mantener información entre diferentes páginas, he de gestionarlo en programación
- Una manera ya le hemos visto usando la función de cabecera

```
header (Location:url?variable1=valor&variable2=valor)
```

Usando campos ocultos

- Otra manera es usando campos ocultos
- Veamos su funcionamiento
- Hay situaciones donde queremos recopilar además de la información que el usuario rellena, algún dato más.
- Supongamos que queremos saber las veces que una página se invoca a sí mismo

- Cualquiera pensaría en crear una variable, y cada vez que llamemos a la página incrementar en una unidad



- Cada vez que llamemos a la página siempre que el usuario se haya identificado vamos a especificar las veces que ha invocado a la página.
- Para ello necesitamos enviar a la página del servidor la información de las veces que se ha invocado a la página

```
<input type="hidden" name="valorOcultoaRescatar" value="$variable">
```

Actividad

Implementa el programa anterior y verifica su funcionamiento

Transfiriendo ficheros entre cliente y servidor

- Es muy sencillo y frecuente subir ficheros entre cliente y servidor

- Cuando vamos a subir ficheros hay que conocer acciones a indicar tanto en la parte de cliente como en la de servidor.

Acciones en el Cliente

input type=file

- Debemos especificar un elemento ***input*** con de **type *file*** en un formulario
- Como todo input debe tener asignado un **name** para acceder a él en el servidor

```
<input type=file name=fichero>
```

form method=POST enctype="multipart/form-data

- El formulario donde esté el **input** ha de tener especificado el atributo ***enctype*** establecido con el valor ***mutipart/form-data***.
- Cuando no especificamos tipo, se asume por defecto el valor **application/x-www-form-urlencoded**. Este valor implica que enviamos texto plano y lo podremos enviar tanto por GET como por POST.
- No obstante si vamos a transferir un fichero no necesariamente de texto ***debemos*** especificarlo estableciendo el valor de ***enctype*** a **mutipart/form-data**. Este valor se emplea para transferir gran cantidad de texto u otros formatos de fichero entre cliente y servidor

Recursos de la Web

https://www.w3.org/TR/1999/REC-html401-19991224/interact/forms.html#form-content-type

enctype es un atributo necesario para especificar el tipo de contenio usado para enviar la información del formulario al servidor

- Necesariamente hemos de usar el método POST para este cometido

```
<form action='' method=POST entype='multipart/form-data'>
....
</form>
```

Establecer tamaño en el cliente

- El tamaño de bytes que vamos a enviar también puede quedar establecido en el cliente, de modo que si el fichero tiene un tamaña mayor, no se envía
- Para esto se establece antes del input file, un input hidden con name = MAX_SIZE_FILE y value el valor del tamaño máximo.
- Este mecanismo no avisa al cliente de nada, simplemente dejará de enviar el fichero al servidor en caso de exceder el tamaño.
- En el servidor se recibirá un error de valor ***2*** o constante UPLOAD_ERR_FORM_SIZE, (Ver código de errores más abajo o en ***http://php.net/manual/es/features.file-upload.errors.php***
- Con todo lo dicho, la especificación en el cliente quedaría

```
 <form action="descarga.php" method="POST" enctype="multipart/form-data">
          <input type="hidden" name="MAX_FILE_SIZE" value="10000000">
          <h3>Selecciona fichero </h3>
          <input type="file" name="fichero" id="" >
          <br />
          <input type="submit" value="Acceder" name="descarga">
      </form>
```

Acciones en el Servidor: $_FILES

- La forma de acceder al input del tipo file que viene del cliente en la solicitud al servidor es a través de la superglobal $_FILES
- Lo primero que deberemos hacer es acceder a este elemento con el nombre del input

```
$fichero = $_FILES['nombre_input_file']
```

- $_FILES es un array asociativo con tantos elementos con input de tipo file vengan del formulario cuyo submit ha generado una solicitud al servidor
- Cada posición a su vez contiene un array asociativo con información de ese fichero alamcenada en 5 componentes

1. ***name*** Nombre del fichero en el cliente
2. ***type*** Tupo de fichero subido
3. ***size*** Tamaño en bytes del fichero
4. ***tpm_name*** Nombre asignado de forma temporal en el servidor
5. ***error*** Error que se haya podido producir o 0 si no ha habido ninguno (Ver tabla más abajo)

Con el fichero que viene del cliente, en el servidor podemos hacer:

1. Capturar el fichero y dejarlo en un directorio concreto
2. Ver si se ha producido algún error especificando el código de error mediante una constante numérica
3. Ver el tamaño del fichero
4. Analizar el tipo de fichero para poder por ejemplo aceptarlo o descartarlo o decidir en qué carpeta dejarlo en función del tipo

Copiando el fichero a una carpeta

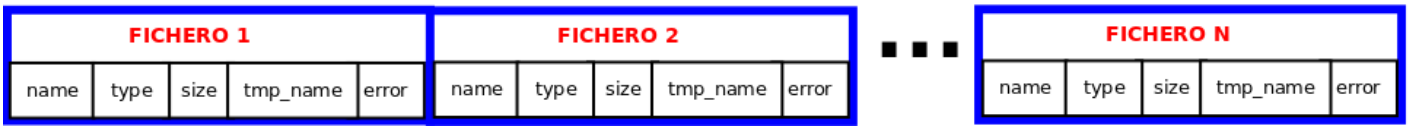
- La primera acción será copiarnos el fichero en una ubicación concreta dentro de nuestro servidor
- Lógicamente primero deberemos crear esa carpeta y asegurarnos que tenga permisos de escritura en ella el usuario apache (normalmente www-data)
- En el servidor tenemos el fichero disponible de forma temporal en la capeta /tmp. Podemos acceder a esta información en el elemento ***$_FILES['tmp_name']***
- Para copiarlo usaremos la función ***move_uploaded_file($origen, $destino);***, dónde ***$origen*** es el fichero que queremos copiar con ubicación y $destino es la ubicación y nombre de fichero donde queremos dejarlo.
- Lo más habitual es dejar el fichero con el mismo nombre que tenía en el cliente, esta información la tenemos disponible en el atributo ***$_FILES['name']***
- La función ***move_upload(..)*** *retorna un booleano que indica el éxito o fracaso de la acción http://php.net/manual/es/function.move-uploaded-file.php.*

- A continuación un resumen de estas acciones

```
//Suponemos en el cliente
//... <input type=file name= fichero>
//
//Accedemos al fichero que está de forma temporal en el servidor
$origen = $_FILES['fichero']['tmp_name'];
//Accedemos al nombre del fichero con el que el cliente lo subió
$nombreFichero = $_FILES['fichero']['name'];
//Establecemos la ruta donde queremos dejar el fichero
//En este caso en la carpeta del proyecto tenemos una carpeta llamada descargas con permiso de escritura para www-data
$destino ="./descargas/".$nombreFichero
//Ahora procedemos a copiar y ver el éxito o fracaso
if (move_uploaded_file($origen, $destino))
   echo ("El fichero $nombreFichero se ha subido correctamente");
else
   echo ("Error subiendo el fichero $nombreFichero");
```

Comprobando errores

- ***$_FILES[error]*** contiene información del error que se ha podido producir al subir el fichero
- La siguiente tabla es la lista de los posibles valores que va a haber en este elemento del array superglobal $_FILES



CÓDIGOS DE ERROR SUBIENDO FICHEROS

Valor entero	Constante	Descripción
0	***UPLOAD_ERR_OK***	Fichero subido exitosamente
1	***UPLOAD_ERR_INI_SIZE***	*Tamaño excedido según directiva **upload_max_filesize** de php.ini.*
2	***UPLOAD_ERR_FORM_SIZE***	*El fichero subido excede la directiva MAX_FILE_SIZE especificada en el formulario HTML.*
3	***UPLOAD_ERR_PARTIAL***	*El fichero fue sólo parcialmente subido.*
4	***UPLOAD_ERR_NO_FILE***	*No se subió ningún fichero.*
6	***UPLOAD_ERR_NO_TMP_DIR***	*Falta la carpeta temporal.*
7	***UPLOAD_ERR_CANT_WRITE***	*No se pudo escribir el fichero en el disco.*
8	***UPLOAD_ERR_EXTENSION***	*Una extensión de PHP detuvo la subida de ficheros.*

Recursos de la Web

http://php.net/manual/es/features.file-upload.errors.php

- Un posible código para obtener esta información

```
//Suponemos en el cliente
//... <input type=file name= fichero>

$fichero = $_FILES['fichero'];
.....
$error = $fichero['error']
//Esto es igual que hacer $error = $_FILES['fichero']['error']

$error = $_FILES['error'];
switch ($error){
    case 0:
        echo "ERROR. Fichero subido de forma correcta. <br />";
        break;
    case 1:
        echo "ERROR. Tamaño de fichero superior al establecido en el servidor <br />";

        break;
    case 2:
        echo "ERROR. Tamaño de fichero superior al establecido en cliente<br />";
        echo "El tamaño se estableció en el input MAX_FILE_SIZE<br/>";
        echo "Tamaño establecido ".$_POST['MAX_FILE_SIZE']."<br/>";
```

```
        break;
    case 3:
        echo "ERROR. EL fichero sólo se subió parcialmente <br/>";
        break;
    case 4:
        echo "ERROR. No se subió ningún fichero <br/>";
        break;
    case 6:
        echo "ERROR. No se encuentra la carpeta temporal <br/>";
        break;
    case 7:
        echo "ERROR. No se pudo escribir en disco. revisa permisos <br/>";
        break;
    case 8:
        echo "ERROR. Una extensión de php detuvo la subida del fichero <br/>";
        break;
    default:
        echo "Valor de error desconocido";
}
```

Ver tamaño del fichero y otras directivas en php.ini

- El tamaño de fichero queda definido en el servidor por la directiva de ***php.ini'***

```
upload_max_filesize=
```

- Otras directivas relacionadas con la descargas de ficheros están establecidas en php.ini
- A continuación se detallan con sus valores por defecto. (Ver el fichero php.ini)

```
;;;;;;;;;;;;;;;;
; File Uploads ;
;;;;;;;;;;;;;;;;

; Whether to allow HTTP file uploads.
; http://php.net/file-uploads
;Comentario: Permite la descarga de ficheros
file_uploads = On

; Temporary directory for HTTP uploaded files (will use system default if not
; specified).
; http://php.net/upload-tmp-dir
;;Comentario : Establece el directorio temporal en el servidor donde se deja temporalmente el fichero subido
;;Si no se especifica se tomará /tmp en linux, o el directorio por defecto que use el SO de forma temporal
;upload_tmp_dir =

; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
;;Comentario : Tamaño máximo del fichero permitido en el servidor
;;Se puede usar los múltiplos K M G T
upload_max_filesize = 20M

; Maximum number of files that can be uploaded via a single request
;;Número máximo de ficheros que se pueden descargar en una sola solicitud httlp
max_file_uploads = 20
```

- Podemos ver el tamaño exacto del fichero subido mediante el elemento size del array

```
//Suponemos en el cliente
//... <input type=file name= fichero>
....
$size = $_FILES['fichero']['size'];
....
```

- Entre otras cosas puede servir para descartar un fichero de menos de un tamaño concreto.

Tipo de fichero

- Este es un atributo importante

1. Analizar el tipo de fichero para poder por ejemplo aceptarlo o descartarlo o decidir en qué carpeta dejarlo en función del tipo

- Para ver el tipo podemos observar la extensión del fichero

```

```

- O bien analizar el tipo MIME que nos viene en ***$_FILES['type']***
- Por ejemplo suponemos que queremos distribuir los ficheros en tres carpetas

1. Los ficheros que contengan imágenes a la carpeta ***./descargas/imagenes/***
2. Los ficheros que contengan música a la carpeta ***./descargas/musica/***
3. El resto de ficheros a la carpeta ***./descargas/otros/***

- En el tipo mime separa el tipo general del fichero con una barra.
- Así los de tipo música o audio sería ***audio/....'***
- Así los de tipo imagen ***image/....'***

.....

- Un posible código sería

```
//Suponemos en el cliente
//... <input type=file name= fichero>

.....
$origen = $_FILES['fichero']['tmp_name'];
$nombreFichero = $_FILES['fichero']['name'];
$tipo = $_FILES['fichero']['type'];
$tipo_fichero = explode('/', $tipo);
switch ($tipo_fichero[0]) {
    case 'audio':
        $dir_destino = "/var/www/descargas/uploads/musica";
        break;
    case 'image':
        $dir_destino = "/var/www/descargas/uploads/imagenes";
        break;
    default:
        $dir_destino = "/var/www/descargas/uploads/otros";
}
destino = $dir_destino . '/' . basename($nombreFichero);
move_uploaded_file($origen, $destino);
```

Tip: la función ***explode***

Esta función rompe una cadena de caracteres en diferentes campos de un array indexado cada vez que encuentre un determinado carácter. Tango el carácter, como la cadena son argumentos pasados a la función.
http://php.net/manual/es/function.explode.php

Trabajar con directorios

- Una vez subidos los ficheros es habitual que se quieran mostrar en la página para que el usuario los pueda ver o acceder a ellos
- Para ello, php tiene una serie de funciones que nos permite interactuar con un directorio y sus ficheros igual que si trabajáramos en un terminal.

PHP: Trabajando con ficheros y directorios

Funciones de directorio

```
http://php.net/manual/es/ref.dir.php
```

Funciones sobre ficheros

```
http://php.net/manual/es/ref.filesystem.php
```

Tip: No olvidar que es ***www-data*** o el propietario del proceso de ***Apache*** quien ha de tener los permisos necesarios para interactuar con los ficheros o directorios

Ejemplo de uso de la clase *Directory*

- El tema de programación orientado a objetos lo veremos mas tarde, pero sabemos que para invocar a un método de un objeto se usa el operador de indirección **->**
- Los métodos los vemos como funciones que son de una clase y los puede invocar un objeto. (Esta idea imprecisa de momento es suficiente).
- ***Directory*** Es una clase especializada en gestionar directorios (los directorios son ficheros igualmente)
- Para inicializarlo podemos usar un alias de ***new Directory()*** llamada ***dir(...)***
- En su invocación pasamos el directorio que queremos ver

```
  $directorio = dir("/var/www/musica/subidas/");
```

- Para leer el contenido de un fichero o un directorio (los archivos que contiene), se usa el método ***read()***
- Una forma de hacerlo es

```
while ($archivo = $directorio -> read()){
      ..................        }
}
```

- Es importante cerrar el fichero (en este caso de tipo directorio), con el método close

```
    $directorio->close();
```

Recursos de la Web

- Página oficial de php para la descarga de ficheros http://php.net/manual/es/features.file-upload.php
- Listado completo de todos los tipos MIME:

```
https://www.sitepoint.com/web-foundations/mime-types-complete-list/
http://www.iana.org/assignments/media-types/media-types.xhtml
```

- Funciones de php para trabajar con ficheros http://www.w3schools.com/php/php_ref_filesystem.asp

Usuario:ManuelRomero/ProgramacionWeb/php/POO/introduccion

De WikiEducator
< Usuario:ManuelRomero | ProgramacionWeb/php | POO

BLOQUE 2 PHP: **PROGRAMACIÓN ORIENTADO A OBJETOS**

¡Construyendo componentes!

PHP Como lenguaje orientado a objetos

TEMA 6: PHP ORIENTADO A OBJETOS

Contenido

Show presentation

Programación orientada a objetos

- En programación *el paradigma imperativo* está basado en **funciones y datos**.
- *El paradigma orientado a objetos* está basado en **Objetos**.
- Los **objetos** son el elemento básico y central de *la programación orientada a objetos (OOP) o (POO)*.
- Podemos hablar de *universo de discurso* como el sistema que queremos automatizar por software
- Un **Objeto** es una entidad (concreta o abstracta) que presenta una actividad en un entorno concreto, en un determinado universo de discurso.

Definición

Objeto Cada elemento activo que identificamos dentro de un determinado universo de discurso.

Serán nuestros componentes software para ensamblar nuestros programas

Ejemplo

En un banco hay cuentas bancarias (objeto)

Las cuentas bancarias se identifican con un número y un titular (nombre, apellido y dni) ***atributos***
Las cuentas se pueden dar de alta, de baja, hacer extracciones e ingresos y transferencias... ***métodos***

- Puede parecer una forma más complicada de programar, pero es una manera de dividir la naturaleza del problema que estamos estudiando en unidades independientes que pueden interactúar entre ellas.
- Cada una de ellas va a tener una identidad propia asignando valores a sus atributos
- Cada una de ellas va a tener un comportamiento concreto que va a ser lo que sabe hacer para que los demás o el programa principal lo utilice

Elementos en la programación orientada a objetos

- De lo dicho anteriormente deducimos que tenemos dos elementos:

1. Los ***atributos*** o características de la clase.
2. Los ***métodos*** o comportamiento de la clase .

- Para crear objetos, previamente hay que **definir su estructura**.
- La definición de la estructura **(atributos y métodos)** de componentes software se llama ***clase***

Clase

La descripción y especificación de componentes software para su posterior uso en los programas

- **Una clase** es la estructura de un tipo concreto de objetos.
- **Los objetos** son elementos concretos en mi sistema. Instancias de la clase en memoria para ser usadas por un programa

Elementos de la POO

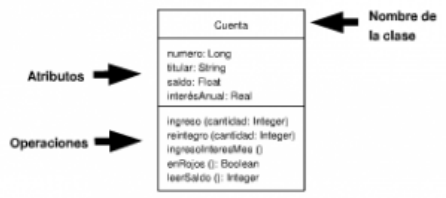
Atributo

- Son las características o datos de un objeto.
- Sus valores nos da el estado de un objeto en un momento dado.
- Normalmente al instanciar un objeto en memoria lo primero que hacemos es dar valores a sus atributos
- Es recomendado que los atributos entén encapsulados solo al objeto (privados)

Elementos de la POO

Métodos

- Especifican el **comportamiento** de los *objetos*.
- Permiten modificar y conocer el estado de un objetos ***(métodos getter and setter)***.
- Permiten que un objeto haga cosas en el sistema *(comunicación entre objetos)* .
- ***Los métodos*** son las acciones que el objeto sabe hacer, ***servicios*** que ofrece
- También son las acciones internas para facilitar las acciones al objeto



OPP En php

- PHP no se diseñó como lenguaje orientado a objetos, por lo que muchas de las características de este paradigma se han ido incorporando en las últimas versiones, especialmente a partir de la versión 5.3.
- PHP Almacena el valor de un objeto como una referencia (dirección de memoria), no guarda el valor.
- Esto implica Insertar aquí texto sin formatoque si queremos pasar un objeto a través de la red, debemos serializarlo, para que *viaje* también el valor del mismo y no solo la dirección de memoria que en destino carecería de sentido. Veremos este concepto más adelante.

En php las clases tienen métodos y propiedades

1. propiedades: son los atributos o características de la clase.
2. métodos: representas el comportamiento de la misma.

Definir una clase en php

```
class NombreClase{
    //propiedades
    //métodos
}
```

- ***NombreClase*** es un identificador válido con la siguiente expresión regular

```
^[a-zA-Z_][a-zA-Z0-9_]*$
```

- El nombre de las clases se recomienda que empiece por mayúsculas
- Es recomendable guardar las clases en ficheros cuyo nombre sea el propio de la clase

Ejemplo

Vamos a crear una clase llamada fecha

- Atributos de la clase (dia, mes, year)
- Métodos *verFecha* (obtener la fecha como una cadena de caracteres)

Tip: A continuación un posible código iremos viendo más adelante muchos detalles sintácticos aquí expresados

```php
<?php

class Fecha {
  private $dia;
  private $mes;
  private $year;

/**
 *
 * @param int $dia
 * @param int $mes
 * @param int $year
 * Método con el nombre de la clase que se ejecuta  cuando se instancia un objeto
 * Recibe tres parámetros con los que inicializa los atributos
 */
  public function Fecha($dia, $mes, $year){
     $this->dia = $dia;
     $this->mes= $mes;
     $this->year= $year;
  }

/**
 * @return string la fecha con formato dd/mm/yyyy
 * Método público para visualizar la fecha
 */
  public function verFecha(){
          return "$this->dia / $this->mes= $mes /$this->year";

  }
?>
```

- En el programa principal

```php
<?php
  require "Fecha.php";
  $f1 = new Fecha(10,12,2016);
  echo "La fecha es ".$f1->verFecha();
// put your code here
?>
```

- Y la salida que se produce

```
La fecha es 10/12/2016
```

- Iremos entendiendo cada parte de esta declaración y uso a lo largo del tema

Pilares básicos de la POO

- Son 4 las características o principios de la programación orientada a objetos

Puntos clave

Encapsulación
Herecia
Polimorfismo
Abstraccion

Encapsulación: Acceso a los componentes

- A la hora de definir tanto las propiedades como los métodos, especificaremos el nivel de acceso que se tiene a ese elemento
- Es una buena práctica de programación no dejar acceso directo a los atributos de una clase, sino acceder a ellos a través de los métodos

Puntos clave

- La encapsulación es uno de los pilares de la programación orientada a objetos

permite o restringe la visibilidad de sus componentes

Visibilidad

- Visibilidad o alcance de las propiedades no constantes y los métodos de las clases
- Implementa el principio de encapsulación.
- Permite especificar el nivel de acceso que se tienen sobre los elementos

Visibilidad

- Son tres los tipos de visibilidad que podemos especificar:

1. public
2. private
3. protected

Tip: ***public*** tipo de visibilidad asignada por defecto en caso de no especificar

public

- Este tipo de visibilidad es asignada por defecto
- Los elementos públicos pueden ser accesibles en cualquier momento del programa.
- Recordemos que para acceder a un elemento debemos especificar el objeto o clase del que queremos el elemento
- Si el elemento es estático o constante usaremos el operador ***::*** llamado operador de especificación ámbito #::
- Si el elemento es no estático accedemos a través del operador **->**

Ejemplo

- En el código anterior ver el método ***verFecha()*** que es **públic**
- Sin embargo las propiedades ***dia***, ***mes***,***year***, son **private**
- Esto implica que en el programa principal puedo hacer

```
....
$f = new Fecha(5,10,2017)
...
$f->verFecha();
...
```

- Pero no puedo hacer

```
....
$f = new Fecha(5,10,2017)
...
$f->dia= 5;
...
```

Tip: En el caso de que las propiedades fueran **public**, sí podría hacerlo

private

- Los elementos especificado con este modificador de acceso hace que su visibilidad se reduzca al interior de la clase, no pudiendo acceder a ellos desde fuera
- En OOP es una tendencia hacer todos los atributos privados y acceder a ellos por los métodos setter and getter.

```php
<?php
class Usuario {
    private $usuario;
    private $password;

    public function __construct($usuario) {
        $this->password = 'passDefecto';
        $this->usuario = $usuario;
    }

    public function __destruct() {
        echo 'Me voy.....';
    }
    public function getUsuario(){
        return $this->usuario;
    }
    public function getPass(){
        return $this->password;

    }
    public function setUsuario($user){
        this->usuario =$user;
    }
    public function setPass($pass){
        this->password =$pass;
    }
}

$usuario = new Usuario('manolo');
//....
$pass = $_POST['pass'];
//....
$usuario->setPass($pass);
//...
?>
```

Tip: A un elemento ***private*** de una clase, tampoco podrá acceder desde clases que deriven de ésta

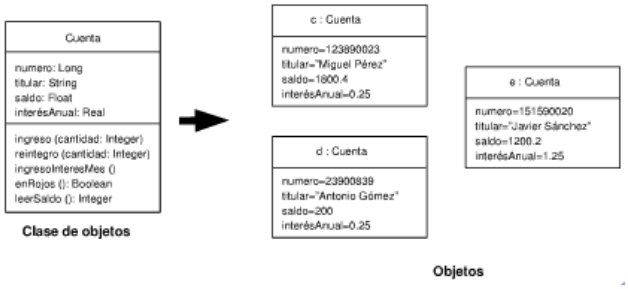
protected

- Este tipo de visibilidad implica que los elementos así especificados solo son accesible por la propia clase y por las clases derivadas
- Para ello hay que ver la herencia que veremos más adelante dónde propondremos un ejemplo

```php
<?php
class persona{
    protected $nombre;
    protected $fNac;
    //....
}
//.....
class medico extends persona{
    private $numColegiado;

    public function __construct($nombre, $fechaNacimiento, $colegiado) {
        $this->nombre=$nombre;
        $this->fNac=$fechaNacimiento;
        $this->numColegiado=$colegiado;

    }
    public function visualiza(){
        echo "Medico $this->nombre";
    }
}
$medicoPueblo1= new medico("pedro", "1/1/1969","123456");
$medicoPueblo1->visualiza();

?>
```

Declarando objetos: Operador ***new***

- Permite crear instancias de un objeto en memoria.
- Una clase describe lo común de unos determinados objetos, la estructura o composición.
- Las clases en principio no se usan durante la ejecución, salvo si queremos acceder a **métodos o propiedades estáticas** como veremos un poco más adelante
- Lo que se usa en los programas son los ***objetos***.
- Para ello debemos *instanciar* objetos de las clases
- Esto se hace con el operador **new**
- Una vez **instanciado** ya tenemos la referencia del objeto y lo podemos utilizar
- hay que pensar que en memoria tenemos **toda** la estructura del la clase por cada objeto



$this

- Accediendo a los atributos de un objeto: seudovariable $this
- ***$this*** es una seudovariable que referencia al objeto del ámbito en el cual está usado
- Se utiliza dentro de la definición de la propia clase y hará referencia a un objeto concreto en un momento dado; esto dentro de la clase en la que está siendo utilizado

Ejemplo

```php
<?php
class MyClase
{
    function ser_estar()
    {
        if (isset($this)) {
            echo '$this Ahora soy por que estoy';
        } else {
            echo "\$this ni es ni está.\n";
        }
    }
}
```

- Podemos probar este código de la siguiente manera en un programa principal

```php
....
        require "MyClass.php";
        $a = new MyClass();
        $a->ser_estar(); //Invocamos al método de un objeto

//Invocamos al método de manera forzada, sin que exista un objeto concreto

        MyClass::ser_estar();
```

self

- Cuando queremos acceder a un elemento estático o una constante, éstos son valores que no se establecen en memoria para cada objeto que declare, sino que son compartidos por todos los objetos de la clase, habiendo en memoria un solo valor de los mismos.

Cuando queremos acceder a ellos dentro de la clase (en la declaración de la estructura), los referenciaremos con el operador ***self'**, que se podría traducir como **yo mismo***

- Por ejemplo si tengo una constante declarada

```php
<?php
class Constantes{
      const K = 10;
      const IVA = 0.21;
      function getValores(){
              echo "Valor de la constante --".self::K."--<br/>";
              echo "Valor del producto de 235 euros base . cuyo iva es ".(self::IVA*235);
      }
}
```

Acceso al contenido del objeto : -> y ::

- Ya hemos visto que para acceder a un elemento de un objeto usamos operadores **->** o bien ***::***

Operador de indirección ->

- Este operador es un operador de indirección
- Los objetos son direcciones de memoria, cuando se quiere acceder al contenido de una dirección de memoria se usa un operador de **indirección**, que en el caso de php como en otros muchos lenguajes es **->** .

- Observar que se suele acompañar de una variable objeto o de la seudovariable ***$this*** , por ese motivo si se quiere acceder a una propiedad del objeto, ya no hay que especificar el ***$*** en el nombre de la propiedad

```
class Clase1{
 public $propiedad1;
 ....
 public function __construct($valor){
//la variable o propiedad de la clase no lleva $ al acceder a ella.
       $this->propiedad1 = $valor;

 }
....
 $obj1 = new Clase1("verde");
 $obj1->propiedad1 ="azul";
...
```

Operador de resolución de ámbito ::

- http://php.net/manual/es/language.oop5.paamayim-nekudotayim.php
- Se utiliza para poder acceder a los elementos estáticos de la clase
- En la parte de la izquierda hay que especificar el dominio o elemento al que pertenece la propiedad o método estático.
- Podremos usar;

1. nombre de clase,
2. nombre del objeto
3. ***self*** : si es dentro de la misma clase
4. ***parent*** : si el elemento pertenece a la clase de la que heredo
5. ***static*** Al igual que self se puede usar la palabra reservada static, para acceder a un elemento estático de la clase.

Resolución de ámbito

El siguiente código aclara de forma completa estas posibilidades

```
class Clase1 {
    //put your code here
    const  IVA = 21;
    public static $numObj ;
    public function __construct() {
        self::$numObj++;
        echo "En total hay ".Clase1::$numObj." objetos de esta clase e IVA = "
                                                    .static::IVA."<br />" ;
    }

}

$obj1 = new Clase1();
$obj2 = new Clase1();
$obj3 = new Clase1();
echo "<hr />";
echo "El valor del atributo estático numObj lo
 puedo ver desde cualquier objeto de la clase <br />";
echo "NumObj desde obj1 ".$obj1::$numObj. "<br />";
echo "NumObj desde obj2 ".$obj2::$numObj. "<br />";
echo "NumObj desde obj3 ".$obj3::$numObj. "<br />";
echo "NumObj desde en nombre de la clase ".Clase1::$numObj. "<br />";
?>
```

- La salida que produciría el código sería

```
En total hay 1 objetos de esta clase e IVA = 21
En total hay 2 objetos de esta clase e IVA = 21
En total hay 3 objetos de esta clase e IVA = 21
El valor del atributo estático numObj lo puedo ver desde cualquier  objeto de la clase
NumObj desde obj1 3
NumObj desde obj2 3
NumObj desde obj3 3
NumObj desde en nombre de la clase 3
```

Propiedades

- Al igual que en el código estructurado los valores que almaceno en memoria, las propiedades de los objetos pueden ser.

1. Variables
2. Constantes

Constantes

- Para definir constantes se usa la palabra reservada ***const***. Como ya sabemos este valor no puede ser modificado durante la ejecución.
- El identificador de las constantes no empieza por $.
- A una constante hay que asignarle un valor no pudiendo asignar expresiones.
- Todos los objetos de la misma clase comparte el valor de la constante. Por lo que se tomará como un valor estático.
- Antes de la versión 7.1, incluyendo la 7.0, las constantes siempre eran públicas
- A partir de la versión 7.1 se puede especificar la visibilidad (public, protected o private)

Accediendo al valor de una constante

1.- Dentro de la clase:

- Operador ***self*** junto con el ***operador de resolución de ámbito ::***
- Nombre de la clase

2.- En el programa:

- Nombre de la clase
- Nombre de cualquier objeto de la clase
  - En ambos casos, junto con el operador de resolución de ámbito ***::***, seguido del identificador de la constante.
- Vemos un ejemplo de su uso

```
<?php
class Constantes{
      const K = 10;
      const IVA = 0.21;
      function getValores(){
              echo "Valor de la constante --".self::K."--<br/>";
              echo "Valor del producto de 235 euros base ".((self::IVA*235)+235);
      }
}

$a=new Constantes();
//Mostramos los valores de las constantes
$a->getValores();

echo "<br/>valor de la constante con el nombre de la clase ".Constantes::K;
echo "<br/>valor de la constante con el nombre del objeto".$a::K;
?>
```

Variables

- Estas propiedades son como las variables pero de la clase.
- Siguen la misma regla de construcción que vistas anteriormente.
- Las propiedades de la clase al igual que los métodos se les puede especificar una determinada #visibilidad o alcance, siendo el valor por defecto ***public***.
- También puedes ser #static o estáticas;Este especificador establece que estos elementos sean conocidas como propiedades o métidos de la clase, si se especifica con la palabra reservada ***#static***.

Tip: Es importante recordar que para acceder dentro de la clase a los métodos o propiedades de ella, hay que usar la seudovariable ***#$this'***

- Esto es debido a que php es de tipado dinámico, si no lo hiciéramos estaríamos accediendo a una variable local al método

Tip: Recordar que en este caso no podremos el ***$*** delante del nombre de la propiedad.

```
<?php
class Propiedades{
     public $propiedad = "rojo";
     public function getPropiedad(){
        echo "\$propiedad ahora es una variable local a método y no tiene valor: --$propiedad--<br/>";
        $propiedad="azul";
        echo "Ahora visualizo el valor de \$propiedad del método:  --$propiedad--<br/>";
        echo "Ahora visualizo el valor de \$propiedad de la clase:  --$this->propiedad--<br/>";
     }
}

$a = new Propiedades();
$a->getPropiedad();
?>
```

Métodos

- Es la forma de especificar el comportamiento de la clase
- Es lo que el objeto va a saber hacer dentro del programa
- Los métodos de detallan usando la palabra reservada ***function***
- En php dentro de la programación orientada a objetos tenemos una serie o tipo de métodos que es muy importante conocer y se llaman #métodos mágicos, que posteriormente estudiaremos.
- Los métodos mágicos son métodos de la clase que son invocados de manera implícita cuando ocurre alguna circunstancia concreto.
- Por ejemplo como vamos a ver en el párrafo siguiente, cuando se instancia un objeto se invoca (si está implementado), al método mágico __construct. a continuación se explica.

métodos contructor y destructor

- En php, al igual que ocurre en Java, podemos tener un método con el mismo nombre que la clase.
- Cuando instanciamos un objeto de una determinada clase, si existe este método se ejecuta y podríamos entenderlo como constructor de la clase.
- Pero realmente el constructor corresponde a un #método mágico llamado ***__construct()'****que es invocado y ejecutado siempre que se instancie un nuevo objeto de la clase (si lo hemos escrito en la clase). En este caso no se ejecutará el método con el nombre de la case si es que existiera.*
- El igual que tenemos un método que se ejecuta cuando instanciamos un objeto de la clase, existe otro #método mágico que se ejecuta siempre que se destruya una instancia de una clase u objeto, y es el método ***__destruct()***
- Las implementaciones de estos dos métodos, lógicamente son libre para cada clase,
- Su invocación es transparente para el programador (esto es cómo ocurre en todos los #métodos mágicos y se realiza siempre respectivamente al crear el objeto, y cuando este es destruido,
- En el caso de ***__construct***, podemos pasarle argumentos, que serían los valores que aportamos al construir un objeto de la case

Usando constructores

```php
class Clase1 {
    //put your code here

    public function Clase1($m){
        echo "Estoy en constructor de Clase1, método Clase1,
           y he recibido el parámetro <strong>$m</strong>";
    }


}
$obj1 = new Clase1("Mensaje pasado al constructor ");
```

- La salida de este código

```
Estoy en constructor de Clase1, métdo Clase1,
y he recibido el parámetro Mensaje pasado al constructor
```

Alternativamente de forma más correcta establecemos el constructor con el método mágico ***__construct()***

```php

class Clase1 {
    //put your code here

    public function __construct($m){
        echo "Estoy en constructor de Clase1, métdo __construct,
              y he recibido el parámetro <strong>$m</strong>";
    }

}

$obj1 = new Clase1("Mensaje pasado al constructor ");
?>
```

- La salida del código anterior

```
Estoy en constructor de Clase1, métdo __construct,
y he recibido el parámetro Mensaje pasado al constructor
```

Tip: El constructor puede recibir parámetros pasados al crear la instancia del objeto con el operador '**new**'

- En caso de tener los dós métodos, se ejecuta ***solamente*** el código del método ***__construct()***

métodos mágicos

- Una serie de métodos cuyos nombres están reservados y se pueden usar con cualquier objeto de cualquier clase.
- Su nombre siempre empieza por __
- Estos métodos que se invocan automáticamente cuando ocurre algo, en php se conocen como métodos mágicos.
- Un ejemplo son el __construct(...) y __destruct(...)

```
http://php.net/manual/es/language.oop5.magic.php
```

- Otro ejemplo importante son los métodos __toString() y __call($function, $paramters)

__toString()

- Este método es invocado si queremos convertir el objeto en un string
- No recibe parámetros , pues no se invoca de forma explícita
- Lo correcto es que retorne un string

```
class Racional {
    //put your code here
    private $num;
    private $den;

    public function __construct($num, $den){
        $this->num = $num;
        $this->den = $den;
    }
    public function __toString(){
        return ($this->num/$this->den);
    }
}

$r1 = new Racional (8,5);
echo "Valor del objeto r1 = $r1";
?>
```

- La salida de este código

```
Valor del objeto r1 = 8/5
```

__call($metodo, $parametros)

- Este método es invocado siempre que invoquemos a un método de la clase que no exista
- Recibe los siguientes parámetros

1.- ***$metodo*** es el nombre del método invocado 1.- ***$parametros*** es un array indexado con la lista de los parámetros con los que invocamos a la función

uso de __call($metodo, $parametros)

```php
class Racional {
    //put your code here
    private $num;
    private $den;

    public function __construct($num, $den){
        $this->num = $num;
        $this->den = $den;
    }
    public function __call($funcion, $argumentos){
            echo "<h2>Has invocado a un método que no existe en esta clase </h2>";
            echo "Nombre de la función <strong>$funcion</strong><br />";
            echo "Lista de parámetros<br />";
            foreach ($argumentos as $param => $valor){

              echo "parámetro <strong>$param</strong> = <strong>".print_r($valor, true).
                   "</strong> <br />";
//Poner en print_r el segundo parámetro a true,
//hace que esa función en lugar de imprimir, retorna el valor.
            }
    }

    }
}
$r1 = new Racional(5,4);
$r1->metodoInventado1(5,4,5,6,7);
$r1->otroMetodoSinParametros();
$r1->otroMetodo([1,2,3],"parametro2", 5,"ultimo parametro");
?>
```

- La salida de este código

Has invocado a un método que no existe en esta clase

Nombre de la función **metodoInventado1**
Lista de parámetros
parámetro **0** = **5**
parámetro **1** = **4**
parámetro **2** = **5**
parámetro **3** = **6**
parámetro **4** = **7**

Has invocado a un método que no existe en esta clase

Nombre de la función **otroMetodoSinParametros**
Lista de parámetros

Has invocado a un método que no existe en esta clase

Nombre de la función **otroMetodo**
Lista de parámetros
parámetro **0** = **Array ([0] => 1 [1] => 2 [2] => 3)**
parámetro **1** = **parametro2**
parámetro **2** = **5**
parámetro **3** = **ultimo parametro**

Sobrecarga

Un concepto muy importante y básico en la programación orientada a objetos. La sobrecarga es una concreción del principio de **polimorfismo**

polimorfismo

- Podemos tener varios métodos con el mimos nombre, pero diferente número de parámetros o con parámetros de diferente tipo
- El tiempo de ejecución se ejecutará uno u otro en función de los parámetros reales que pasemos en la invocación del método

- Sin embargo este aspecto en php no es del todo intuitivo. No existe la sobrecarga como la entendemos en otros lenguajes.
- No obstante tenemos técnicas para poder simular la sobrecarga.

- Muchas veces es fundamental. Especialmente importante a la hora de sobrecargar el constructor de la clase.
- Para simular la sobrecarga en ***php***, jugamos con el concepto de que una variable que no tenga valor se considera de tipo ***null***.
- Lo vemos con una serie de ejemplos para dejar claro este concepto.
- Tomamos como ejemplo una función:

Ejemplo

```
function verTipoParametros($a,$b,$c){
    echo "Primer parámetro ";
    var_dump($a);
    echo "Segundo parámetro ";
    var_dump($b);
    echo "Tercer parámetro ";
    var_dump($c);
}
```

- Ahora la invocamos de diferente manera y vemos el resultado:

Invocar sin parámetros reales

```
echo "Invocando a <strong>verTipoParametros ()</strong><hr / />";
verTipoParametros ();
}
```

- A pesar de que tiene tres parámetros, la invocamos sin parámetros
- El resultado será que cada parámetro al ejecutar la función será de tipo ***null*** con valor ***null*** (es un tipo válido en php):

Invocando a **verTipoParametros ()**

Primer parámetro

/var/www/expresionRegular/index.php:76:null

Segundo parámetro

/var/www/expresionRegular/index.php:78:null

Tercer parámetro

/var/www/expresionRegular/index.php:80:null

Invocar con 1 parámetro real

```
...
echo "Invocando a <strong>verTipoParametros (5)</strong><hr />";
verTipoParametros (5);
...
```

- En esta caso invocamos con un solo parámetro de tipo entero
- Podemos ver el resultado

Invocando a **verTipoParametros (5)**

Primer parámetro

```
/var/www/expresionRegular/index.php:76:int 5
```

Segundo parámetro

```
/var/www/expresionRegular/index.php:78:null
```

Tercer parámetro

```
/var/www/expresionRegular/index.php:80:null
```

Invocar con 2 parámetros reales

```
  ...
echo "Invocando a <strong>verTipoParametros (5,7)</strong><hr />";
verTipoParametros (5,7);
  ...
```

- En esta caso invocamos con dos parámetros de tipo entero
- Al igual que en caso anterior los parámetros en la función serían 3, dos de ellos con valor de tipo entero, y el tercero con valor y tipo null
- Podemos ver el resultado

Invocando a **verTipoParametros (5,7)**

Primer parámetro

```
/var/www/expresionRegular/index.php:76:int 5
```

Segundo parámetro

```
/var/www/expresionRegular/index.php:78:int 7
```

Tercer parámetro

```
/var/www/expresionRegular/index.php:80:null
```

Invocar con 3 parámetro reales

```
  ...
echo "Invocando a <strong>verTipoParametros ('pedro',5,9)</strong><hr />";
verTipoParametros ('pedro',5,9);
  ...
```

- Ahora pasamos tres parámetros, como vemos el primero de tipo string y los otros dos enteros
- Podemos ver el resultado

Invocando a **verTipoParametros ('pedro',5,9)**

Primer parámetro

LibreOffice Calc

```
/var/www/expresionRegular/index.php:76:string 'pedro' (length=5)
```

Segundo parámetro

```
/var/www/expresionRegular/index.php:78:int 5
```

Tercer parámetro

```
/var/www/expresionRegular/index.php:80:int 9
```

Invocar con 3 parámetro reales uno de ellos un array

```
  ...
echo "Invocando a <strong>verTipoParametros ([1,4,'maría'],true,'sonia')</strong><hr />";
verTipoParametros ([1,4,'maría'],true,'sonia');
  ...
```

- Ahora igualmente pasamos tres parámetros, pero uno de ellos es un array
- Podemos ver el resultado

Invocando a **verTipoParametros ([1,4,'maria'],true,'sonia')**

Primer parámetro

```
/var/www/expresionRegular/index.php:76:
array (size=3)
  0 => int 1
  1 => int 4
  2 => string 'maria' (length=5)
```

Segundo parámetro

```
/var/www/expresionRegular/index.php:78:boolean true
```

Tercer parámetro

```
/var/www/expresionRegular/index.php:80:string 'sonia' (length=5)
```

Sobrecargando el constructor

- Usando esta forma de trabajar vamos a sobre cargar el constructor de una clase
- Tomamos una clase de tipo ***Racional***. Un número Racional es un objeto que tendrá numerador y denominador

- Ahora a la hora de construir el objeto planteamos la posibilidad de poderñp instanciar de la siguiente manera:

```
$r1 = new Racional ("8/5");/*   8/5  */
$r2 = new Racional (5,4);  /*   5/6  */
$r3 = new Racional (5);    /*   5/1  */
$r4 = new Racional ();     /*   1/1   */
```

- Aquí vemos claramente que necesitamos sobrecargar el constructor para que pueda responder a todas las situaciones
- Aplicando los conceptos vistos antes, lo único que tenemos que hacer en el constructor es ir viendo ***de qué tipo'*** *son los parámetros.*

Tip: Recordar que null también es un tipo

- Vemos que podemos tener 0, 1 o 2 parámetros
- Por lo tanto la función constructora tendrá que tener 2 parámetros

```
public function __construct($num, $den) {
 ....
}
```

- Especificamos el código do cómo se podría hacer

```
class Racional {

    private $num;
    private $den;
```

```php
    public function __construct($num, $den) {
        //opciones new Racional () =>1/1
        //opciones new Racional (5) =>5/1
        //opciones new Racional ("5/2") =>5/2
        ////opciones new Racional (5,2) =>5/2
        //Otra sitiación no se instancia
        if (is_null($den)) {
            switch (is_numeric($num)) {
                case true:
                    $this->racionalNum($num);
                    break;
                case false:
                    if (is_null($num)){
                        $this->racionalVacio();
                        break;
                    }
                     else {
                        $this->racionalCadena($num);
                        break;
                     }
            }
        }else {
            $this->racionalNumDen($num,$den);
        }


    /*Método para visualizar el objeto como cadena de caracteres*/
    public function __toString() {
        return ($this->num."/".$this->den);
    }

    //A continuación los métodos privados para asignar valores
    private function racionalNum($num) {
        $this->num = $num;
        $this->den = 1;
    }
    /**
     *
     * @param string $num numero racional del tipo "a/b"
     * hay muchas forma de poder descomponer ese array en dos números
     */
    public function racionalCadena($num) {
        $this->num = (int) $num;
        $this->den = substr($num, strpos($num, "/") + 1);
    }
    public function racionalVacio() {
        $this->num = 1;
        $this->den = 1;
    }

    /*En este caso si los valores son incorrectos asigno el racional 1/1  */
    public function racionalNumDen($num, $den) {
        if (is_numeric ($num) && is_numeric($den)){
            $this->num = $num;
            $this->den = $den;
        }else{
            $this->num = 1;
            $this->den = 1;
        }
    }
}
```

- Probamos este constructor con el siguiente código

```php
$a= new Racional();
$b= new Racional(5);
$c= new Racional(5,6);
$d= new Racional("6/6");

echo "Valor del racional \$a  = $a <br />";
echo "Valor del racional \$b  = $b <br />";
echo "Valor del racional \$c  = $c <br />";
echo "Valor del racional \$d  = $d <br />";
```

- Mostrando los siguientes resultados

```
Valor del racional $a = 1/1
Valor del racional $b = 5/1
Valor del racional $c = 5/6
Valor del racional $d = 6/6
```

Sobrecarga con __call(...)

- Otra forma de poder hacer lo mismo es usando el método magico __call($funcion, $parametros).

Tip: El método mágico __call(..) es ejecutado cuando invocamos a un método que no existe en la clase

- Ahora queremos usar un método llamado ***asigna*** que nos permita cambiar el valor de un racional. La forma de aportar el nuevo valor, queremos que sea la misma que la forma de construir el objeto

```php
$r1 = new Racional (); //construye el objeto 1/1

$r1->asigna("6/4"); //Ahora el objeto vale 6/4
$r1->asigna(); //Ahora el objeto vale 1/1
$r1->asigna(8);//Ahora el objeto vale 8/1
$r1->asigna(124, 6);//Ahora el objeto vale 124/6
```

- La forma de proceder será usando los métodos privados creados anteriormente
- El siguiente código implementa la solución

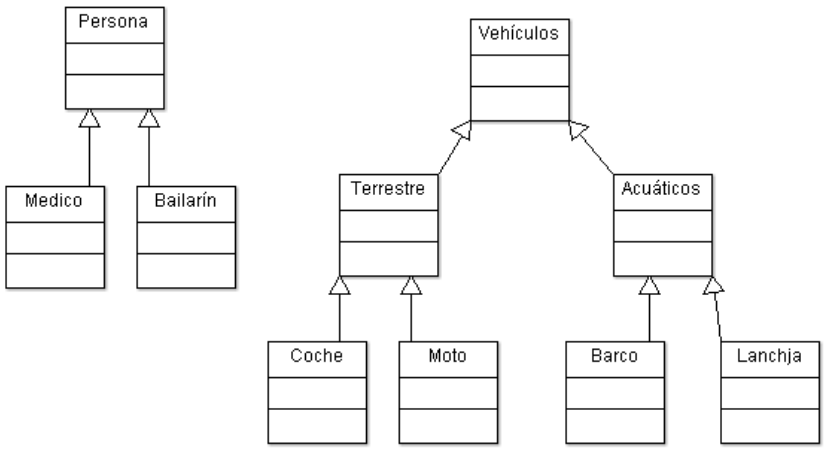
```
public function __call($metodo, $argumentos) {
        if ($metodo == "asigna"){
            switch (count($argumentos)){
                case 0:
                    $this->racionalVacio();
                    break;
                case 2:
                    $this->racionalNumDen($argumentos[0], $argumentos[1]);
                    break;
                case 1:
                    if (is_int($argumentos[0]))
                        $this->racionalNum($argumentos[0]);
                    else
                        $this->racionalCadena($argumentos[0]);
                    break;
            }
        }
    }
```

Herencia

- La herencia es un mecanismo de programación que me permite crear una jerarquía en los componentes software, que se pueden ir especializando

Se puede definir una clase con ciertas características (atributos, métodos)

- Posteriormente puedo definir otra clase a partir de la ya existente, quedándo implícitamente los atributos y métodos como también parte de su estructura o composición
- Es una característica muy natural (p.e Personas (médicos y bailarines) vehículos (Terrestres (coche, moto) Acuáticos (barco, lancha))



- Es una forma de obtener características comunes por separado y luego especializar evitando redundancias
- Facilita la reusabilidad y adaptación
- Vemos dos ejemplos para explicar de forma empírica este concepto

Herencia: gestión personal ambulatorio

- Se pide gestionar un ambulatorio.
- Para ello vamos a hacer sólo el diagrama de clases y su implementación.
- Lo hacemos a nivel básico (sin entrar en detalles).
- Tras realizar un análisis se determina que se pretende gestionar:

los datos de los empleados y anotar las acciones básicas que realizan.

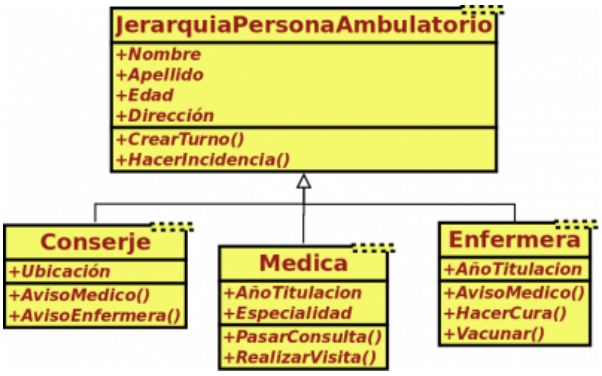
- Encontramos los siguientes elementos que especificamos como clases

Conserjes
Enfermeras
Médicas

```
Las propiedades (atributos) y métodos de cada clase se especifican
en los siguientes diagramas de clases
```

Clase Conserje Clase Enfermera Clase Medicas

- Claramente vemos que todos ellos tienen varios elementos en común.
- Esto nos permite crear una clases genérica que por ejemplo podemos llamar ***personalAmbulatorio***
- Posteriormente creamos una especialización de ***personalAmbulatorio*** con los elementos particulares
- El digrama podría quedar



Posible Solución

clases abstractas

- Es una situación particular que se presentan en muchas jerarquías
- Cuando realizamos jerarquías muchas veces encontramos métodos comunes a varias clases. Esto implicaría que ese método sería un método de una superclase o clase padre de la que luego se heredará
- Pero puede ocurrir que aunque el concepto del método es común a todas las clases,la forma de implementarla es particular en cada una de ellas.
- En este caso, la forma correcta de proceder, es especificar el método el la clase superior, e implementar el código en cada una de las clases que derivo.
- El método especificado en la clase superior sería un método sin código, conocido como un método abstracto, y la clase dónde se especifica pasa a ser abstracta

Clase Abstracta

Es aquella clase que tiene un método o más abstracto

Método Abstracto

- Es un método que no tiene código asociado
- El código se implementará en las clases derivadas

Puntos clave

Nunca podremos instanciar un objeto de una clase abstracta

- Esto es normal, ya que ese objeto no tendría instrucciones para su/s método/s abstracto/s

- Vamos a platear un ejemplo

App de Geometría

Gestionar figuras geométricas de tipo triángulo, cuadrado y Rectángulo.

- De ellas queremos conocer:

1. el número de lados.
2. calcular el área.
3. dibujar el polígono (Esto insertaremos un pequeño código de javascript de uso básico del canvas
4. conocer el número de lados del polígono.

- Realizamos los siguientes clases

Clase Polígono

Descripción de la Clase Polígono

Clase Rectángulo

Descripción de la Clase Rectángulo

Clase Cuadrado

Descripción de la Clase Cuadrado

Clase Triangulo

Descripción de la Clase Triángulo

Uso de esta aplicación

- Un posible código index.php que lo único que hace es crear objetos y visulizar su área y dibujarlos

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        spl_autoload_register(function ($clase){
            require "$clase.php";
        });

        $triangulo = new Triangulo(200,300);
        $cuadrado = new Cuadrado(200);
        $rectangulo = new Rectangulo(200,100);

        echo "<h4>Triángulo,  ". Poligono::lados($triangulo)." y área ".
                $triangulo->area()." px<sup>2</sup></h4>";

        echo "<h4>Cuadrado, ". Poligono::lados($cuadrado)." y área ".
                $cuadrado->area()." px<sup>2</sup></h4>";
        echo "<h4>Rectángulo, ". Poligono::lados($rectangulo)." y área ".
                $rectangulo->area()." px<sup>2</sup></h4>";

        echo $triangulo->dibuja();
        echo $cuadrado->dibuja();
        echo $rectangulo->dibuja();


        ?>
```

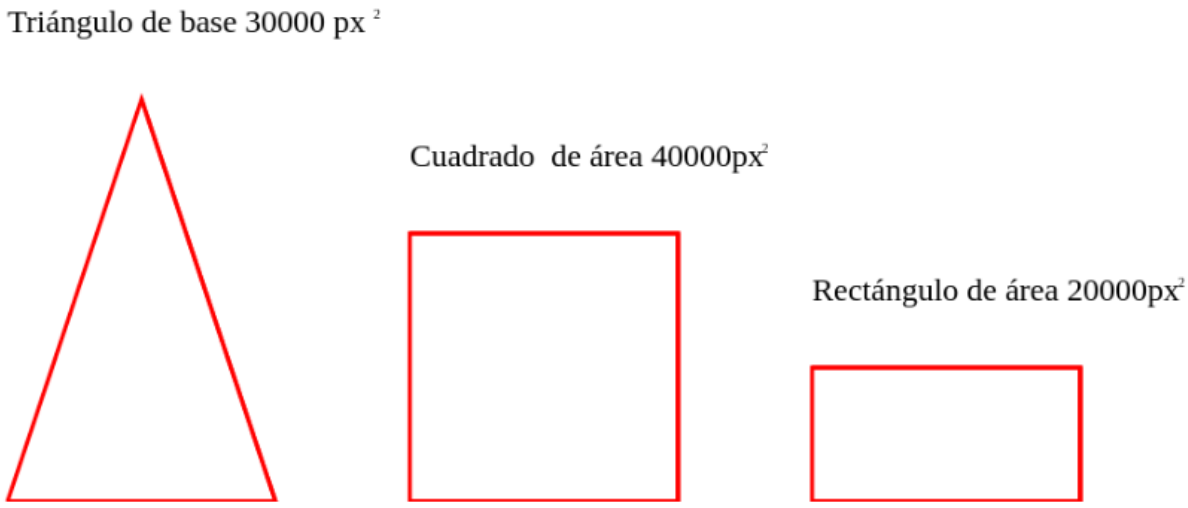
```
    </body>
</html>
```

Resultado de su ejecución

Triángulo, Polígono de 3 lados y área 30000 px^2

Cuadrado, Polígono de 4 lados y área 40000 px^2

Rectángulo, Polígono de 4 lados y área 20000 px^2



V

Expresiones Regulares (er)

Una expresión regular consiste en establecer un patrón o conjunto de caracteres de manera general. De esta forma podremos comprobar si una determinada expresión compuesta por una serie de caracteres concretos, cumple o no la expresión regular
Esta es una técnica extremadamente útil para verificar cualquier tipo de cadena de caracteres, por ejemplo teléfono, email, url, etc.
Es importante tener claro que una expresión regular la podemos utilizar para ver si una cadena coincide exactamente con el patrón, o solo lo continene (En este caso podría ser que empezara por una expresión o que terminara por ella, o que estuviera dentro de ella).
Antes de poder ver algún ejemplo, necesitamos saber cómo se expresa una er en php.

Delimitadores

- A la hora de expresar una expresión regular debemos de hacerlo usando un carácter delimitador al comienzo y final; Este carácter es de elección libre

Delimitadores

http://php.net/manual/es/regexp.reference.delimiters.php

```
/una expresión/   //Carácter delimitador /
#^[^0-9]$#        //Carácter delimitador #
+php+             //Carácter delimitador +
%[a-zA-Z0-9_-]%   //Carácter delimitador %
```

Expresando la expresión regular

- Para especificar las expresiones regulares se puede especificar el carácter tal cual, o usar metacaracteres y agrupamiento de caracteres
- Ver la siguiente tabla sacada de la dirección http://www.mclibre.org/consultar/php/lecciones/php_expresiones_regulares.html

Patrón	Significado
c	carácter c
.	-
^c	empezar por el carácter c
c$	terminar por el carácter c
c+	1 o más caracteres c
c*	0 o más caracteres c
c?	0 o 1 caracteres c
\n	nueva línea
\t	tabulador
\	escape, para escribir delante de caracteres especiales: ^ . [ ] % ( ) | * ? { } \
(cd)	caracteres c y d agrupados
c|d	carácter c o d
c{n}	n veces el carácter c
c{n,}	n o más caracteres c
c{n,m}	desde n hasta m caracteres c
[a-z]	cualquier letra minúscula
[A-Z]	cualquier letra mayúscula
[0-9]	cualquier dígito
[cde]	cualquiera de los caracteres c, d o e
[c-f]	cualquier letra entre c y f (es decir, c, d, e o f)
[^c]	que no esté el carácter c
[[:alnum:]]	cualquier letra o dígito
[[:alpha:]]	cualquier letra
[[:digit:]]	cualquier dígito
[[:lower:]]	cualquier letra minúscula
[[:punct:]]	cualquier marca de puntuación
[[:space:]]	cualquier espacio en blanco
[[:upper:]]	cualquier letra mayúscula

Expresiones regulares en php

Ŝon varias las funciones relacionadas con expresiones regulares en php

```
http://php.net/manual/es/book.pcre.php
```

- Usaremos la función ***preg_match($expresion, $cadena)*** para comprobar que una cadena cumple una determinada expresión regular

```
http://php.net/manual/es/function.preg-match.php
```

Coincidencia exacta o solo qué contenga

Es muy importante diferenciar entre que una cadena coincida exactamente con una expresión regular, y que una cadena cumpla una determinada expresión regular.
Para aclarar este concepto vamos a ver el siguiente ejemplo:

```
La siguiente expresión regular establece una cadena que contenga 1 o más números.
```

```
$exp = /[0-9]+/
```

- Observamos las siguientes cadenas

```
$cad1 = "asdfljoiekalasdf"; //Cumple por que tiene un número
$cad2 = "1"; //Cumple por que tiene un número
$cad3 = "134124"; //Cumple por que tiene un número
$cad4 = "asdfasdf4"; //Cumple por que tiene un número
$cad5 = "asdfasd"; //No Cumple por que no  tiene un número
```

Sin embargo si se quiere especificar, que la cadena sólo contenga números, una forma de especificarlo (por supuesto hay más), estableceremos que comience y termine por números (signo **^** para el comienzo y **$** para la terminación).

```
$exp = /^[0-9]+$/
```

De esta forma, en los ejemplos anteriores, solo la cadena almacenada en la variable $cad3, cumple la expresión regular.

Recursos de la Web

1. http://www.mclibre.org/consultar/php/lecciones/php_expresiones_regulares.html
2. http://php.net/manual/es/book.pcre.php

Validar expresiones regulares

- Realiza un programa que permita insertar una expresión regular y una cadena
- Posteriormente hacemos que valide a ver si la cadena cumple o no la expresión regular

Ver el ejercicio funcionando

```
http://manuel.infenlaces.com/dwes/Expresiones_Regulares/
```

Validar expresiones Regulares (Código) [▼]

Usuario:ManuelRomero/ProgramacionWeb/php/Aut Ses Coo/Autentificacion

De WikiEducator
< Usuario:ManuelRomero | ProgramacionWeb/php

BLOQUE 2 PHP: **PROGRAMACIÓN ORIENTADO A OBJETOS**

¡Construyendo componentes!

PHP Como lenguaje orientado a objetos

cpifp Los Enlaces

TEMA 7.1 Autentificación

Contenido

Qué es autentificarse

- Por autentificarse vamos a entender un mecanismo por el cual el servidor web puede estar relativamente confiado en que está siendo consultado por una determinada máquina y/o persona.
- Por ejemplo podemos visitar esta página y ver diferentes modos en los que piden autentificarse

```
https://www.tractis.com/login
```

- Modos de autentificarse

1. Contraseña y usuario
    1. Almacenar el usuario en sesión (Lo veremos más adelante)
2. Dni digital
3. Certificados digitales de usuario
4. Autentificación basada en Tokens
    1. Usando el protocolo Auth 2.0 (Lo veremos en app híbridas usando el protocolo Auth 2.0)
    2. Usando JSON Web Token (JWT) http://self-issued.info/docs/draft-ietf-oauth-json-web-token.html

Sistema de autentificación por token usando usuarios de redes sociales o de la web

1.Autentificarse a través de la cuenta de Twiter

http://www.maestrosdelweb.com/twitter-autenticacion-oauth-api-login/

2.Autentificarse a través de la cuenta de Facebook

http://www.elwebmaster.com/articulos/autenticacion-de-usuarios-con-facebook-connect-y-google-friend-connect

3.Autentificarse a través de la cuenta de Google

http://www.ladrupalera.com/drupal/desarrollo/javascript/como-usar-una-api-de-google-con-autenticacion-traves-de-oauth2

- Seguro que hay mas sistemas y seguro que muy interesantes, pero estos son los que aquí veremos.
- Nosotros en este tema usaremos el modo de contraseña y usuario. En dos temas posteriores, cuando veamos servicios web usaremos otro sistema como google, igualmente se podría usar la cuenta de facebook o twiter para identificarte, en la relación anterior hay algún enlace por si es de tu interés.

La responsabilidad de exigir una clave de acceso puede recaer sobre el servidor web con los módulos de seguridad

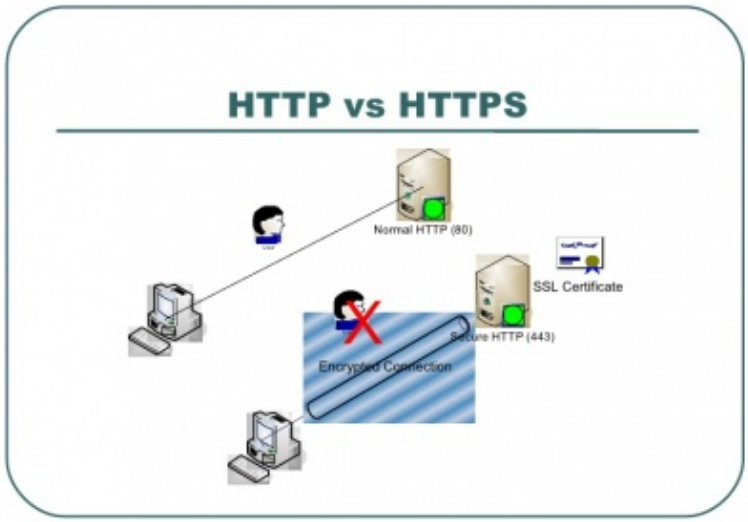
- En este caso restringimos el acceso a la página o sitio web
- Otra opción habitual es tener un sitio con cierto contenido y en el sitio dar la posibilidad de identificarse / registrarse.
- A usuarios identificados se les ofrece otro contenido diferente.

Respecto a estos conceptos vamos a trabajar este tema que es el primero de tres aspectos de este tema

Protocolo http vs https

Este es un concepto importante, diferenciar en contenidos que se envíen de forma segura (usar cifrado en el envío que es lo que se hace con https) y en el hecho que para acceder a un sitio tengan un nivel de seguridad que implique que de alguna manera te tengas que autentificar.
La seguridad del envío de datos es otro aspecto diferente del tema de la autentificación.
En seguridad entre otros aspectos tenemos:

1. ***La autentificación*** mecanismos por los cuales podemos confiar en que quien se ha identificado es conocido para el sistema.
2. ***La confidencialidad*** son mecanismos con los que podemos confiar en que nadie puede ver el contenidio de la información ni modificarla durante la transmisión.



Ambos mecanismos debería de trabajar conjuntamente. Por ejemplo si yo envío una contraseña para identificarse, pero con un **sniffer**, alguien la puede capturar y ver en claro, no conseguimos nada de seguridad.

- Para la transmisión segura se emplea el protocolo https.
- Este tema se estudia en el módulo de despliegue de aplicaciones web.

Tip: Nosotros usaremos http sin cifrar, pero se insiste en que no es segura la transmisión, pues podría ser objetivo de espía informático.

Autentificación por el servidor web

- Es el propio servidor ***http*** quien nos ofrece este método de autentificación.
- Este tema se ve dentro del módulo de despliegue de aplicaciones web, no obstante aquí nos va a interesar, cómo podemos hacer que sea el servidor el que solicite las credenciales y recoja los datos aportados, y cómo desde php podemos ver esos valores o datos y gestionarlos en nuestro programa.
- Recordamos el mecanismo de autentificación que usa ***apache***.

1. Definir los usuarios con acceso permitido

2. Se puede indicar a qué recursos tiene acceso el usuario en concreto (me refiero a qué páginas).
3. Generar en cabecera **http** un código ***http 401*** que es un código de acceso restringido. Este código hace que el usuario reciba un formulario para aportar sus credendiales
4. El navegador al recibir ese código solicitando credenciales
5. El servidor recibe estas credenciales y las almacena en sus variables superglobales para futuras solicitudes

Crear la lista de usuarios

- Usamos la herramienta ***htpasswd*** para crear un fichero con los usuarios y sus contraseñas

```
http://httpd.apache.org/docs/2.4/es/howto/auth.html
```

- En caso de no tener la herramienta instalada debemos hacer la instalación de las herramientas o utilidades de apache escribiendo

```
sudo apt-get install apache2-utils
```

- Este comando tiene una serie de opciones que podemos ver en línea de comandos sin mas que escribir su nombre.

```
Usage:
        htpasswd [-cmdpsD] passwordfile username
        htpasswd -b[cmdpsD] passwordfile username password

        htpasswd -n[mdps] username
        htpasswd -nb[mdps] username password
 -c  Create a new file.
 -n  Don\'t update file; display results on stdout.
 -m  Force MD5 encryption of the password (default).
 -d  Force CRYPT encryption of the password.
 -p  Do not encrypt the password (plaintext).
 -s  Force SHA encryption of the password.
 -b  Use the password from the command line rather than prompting for it.
 -D  Delete the specified user.
On other systems than Windows, NetWare and TPF the '-p' flag will probably not work.
The SHA algorithm does not use a salt and is less secure than the MD5 algorithm.
```

- Es importante la opción ***-c*** para crear el fichero. Recuerda usarla sólo la primera vez; cuando la usas se crea el fichero, y si ya existiera se elimina el contenido del fichero y se crea de nuevo.
- Para incorporar nuevos usuarios se escribe sin opción y se añade el nombre de los usuario
- Por seguridad es importante añadir el fichero en una ubicación fuera del directorio ***documentRoot*** del servidor web.

Prueba a hacer la actividad 1

Actividad 1

Tip: Click con botón derecho del ratón lo abres en una nueva ventana

Indicar los recursos restringidos : .htaccess

- Para este cometido, **apache** nos permite usar el famoso fichero ***.htaccess*** que a continuación vamos a explicar.
- Para habilitar el uso de este fichero debemos indicarle al servidor web que vamos a utilizarlo. Esto se especifica en la directiva ***AllowOverride***
- Para más información ver

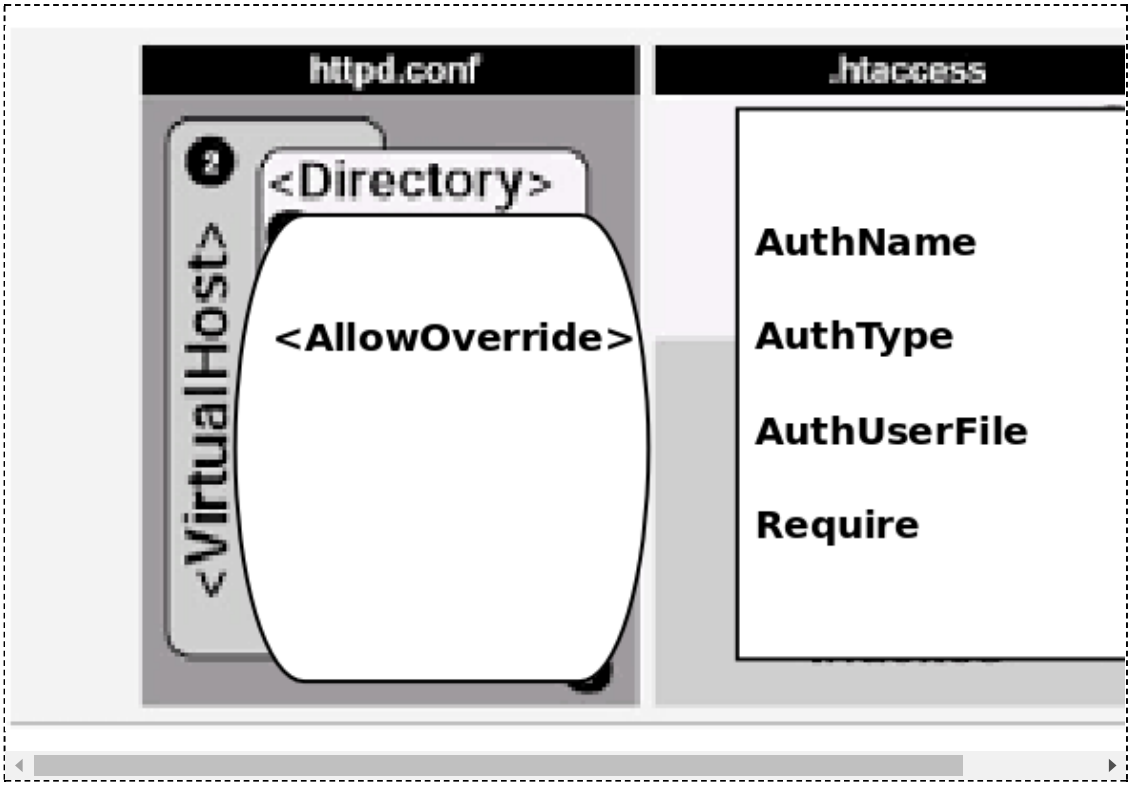
```
http://www.bdat.net/documentos/apache/x367.html
```

Y la oficial (siempre la mejor).

```
http://httpd.apache.org/docs/2.4/es/mod/core.html#allowoverride
```

- En nuestro caso tenemos que poner el valor ***AuthConfig*** o bien ***All***

- Una vez que hemos hecho esto, cada vez que vaya a coger un fichero de un determinado directorio, antes de entregarlo, verificará que si existe un fichero ***.htaccess*** en ese directorio, en cuyo caso pedirá credenciales.
- Para indicar los recursos restringidos usaremos las siguientes directivas (Este será el contenido del fichero .htaccess
- ***AuthName*** Nombre de dominio de la authentificación
- ***AuthType*** Tipo de autentificación, pudiendo ser **Basic** y más segura **Dijest**
- ***AuthUserFile*** Ruta del fichero de los usuarios con permiso
- ***Require valid-user*** o usuarios concretos de la lista que sí que tendrán acceso al recursos si se acreditan correctamente.



Accesos restringidos

Accesos Restringidos

Tip: Click con botón derecho del ratón lo abres en una nueva ventana

PHP accediendo a información http

- Ahora viene la parte que realmente nos interesa a nosotros/as, qué es usar esto dentro de nuestro código ***php***.
- Desde el código de php podemos acceder a la información facilitada por el servidor.
- La matriz asociativa variable superglobal ***$_SERVER*** contiene esta información.

C*Concretamente los índices

1. ***PHP_AUTH_USER*** Es el nombre del usuario
2. ***PHP_AUTH_PW*** Es la password del usuario
3. ***AUTH_TYPE*** Es el tipo de seguridad utilizado

PHP accediendo a los datos de identificación

Php accediendo a los datos de identificación

Tip: Click con botón derecho del ratón lo abres en una nueva ventana

Usar función ***header*** para la autentificación

La referencia de la función ***header*** Sirve para crear cabeceras de la página solicitada

```
http://es.php.net/manual/es/function.header.php
```

Vamos a usarla para unas acciones concretas.

Modificar el texto de error

En este caso no vamos a usar el fichero ***.htaccess***. Cuando accedemos a una página y no nos hemos identificado aparece un error 401. Nosotros vamos a solicitar una autentifiCación ***tipo basic***. < br/> Para ello especificamos la cabecera ***WWW-Authenticate***.Ademas de especificar el modo de autentificación ponemos el mensaje que verá el usuario:

```
    header('WWW-Authenticate: Basic Realm="Página de acceso restringido. Necesitas credencia
```

Para no mostrar la página solicitada se envía un código 401, lo que provoca que se solicite un usuario y contraseña antes de visualizar el resto de la página. El código quedaría:

```
    header('HTTP/1.0 401 Unauthorized');
```

El código completo quedaría:

```
 <?php
if (!isset($_SERVER['PHP_AUTH_USER'])) {
    header('WWW-Authenticate: Basic Realm="Contenido restringido"');
    header('HTTP/1.0 401 Unauthorized');
    echo "Usuario no reconocido!";
    exit;
}
?>
```

Ahora debemos poder especificar cuál va a ser el usuario esperado en esta página. Por ejemplo en la página anterior insertamos un usuario cualquiera y una password cualquiera entraremos ya que no comparamos con ningún valor.

```
<?php
    echo"Nombre de usuario: ".$_SERVER['PHP_AUTH_USER'] ."<br />";
    echo"Contraseña: ".$_SERVER['PHP_AUTH_PW'] ."<br />";
?>
```

Verificar el usuario y contraseña

Con el siguiente código verificaríamos el usuario y contraseña usando el modo de ***header***

```
<?php
if ($_SERVER['PHP_AUTH_USER']!='manolo' || $_SERVER['PHP_AUTH_PW']!='passManolo.') {
    header('WWW-Authenticate: Basic Realm="Contenido restringido"');
    header('HTTP/1.0 401 Unauthorized');
    echo "Usuario no reconocido!";
    exit;
}
?>
```

Esto nos podría servir, puesto que el código no se envía al cliente, y él nunca podría ver la password, ni el usuario. No obstante, siempre será una opción más segura, y más versátil tener el usuario almacenado en una base de datos que ya veremos más adelante cuando veamos bases de datos.

Controlando accesos

Controlando accesos

Tip: Click con botón derecho del ratón lo abres en una nueva ventana

---

Usuario:ManuelRomero/ProgramacionWeb/php/Aut Ses Coo/Sesiones

De WikiEducator
< Usuario:ManuelRomero | ProgramacionWeb/php

BLOQUE 2 PHP: **PROGRAMACIÓN ORIENTADO A OBJETOS**

¡Construyendo componentes!

PHP Como lenguaje orientado a objetos

Autentificación | **Sesiones** | Cookies | Ejercicios | Práctica | Volver

Contenido

¿Qué son las sesiones?

- Es una forma de hacer que variables estén disponibles en múltiples páginas

Por qué necesitamos sesiones

Porque la programación web está basado en http, un protocolo sin estado.

Ideas generales de las sesiones

- A diferencia de las cookies, las variables de sesión se almacenan en el servidor

Cookies

Sabemos qué son las cookies

- Tienen un tiempo limitado de existencia.
- Para identificar al usuario que generó las variables de sesión, el servidor genera una clave única que es enviada al navegador y almace cookie.
- Luego, cada vez que el navegador solicita otra página al mismo sitio, envía esta cookie (clave única) con la cual el servidor identifica d proviene la petición y puede rescatar de un archivo de texto las variables de sesión que se han creado.
- Cuando han pasado 20 minutos sin peticiones por parte de un cliente (navegador) las variables de sesión son eliminadas automáticam configurar el entorno de PHP para variar este tiempo).
- Una variable de sesión es más segura que una cookie ya que se almacena en el servidor.
- No tiene que estar enviándose continuamente como sucede con las cookies.
- Cuando el navegador del cliente está configurado para desactivar las cookies las variables de sesión, tienen forma de funcionar (enviar como parámetro en cada hipervínculo).
- Desventaja: ocupa espacio en el servidor.

Dónde se almacenan

Las variables de sesión se almacenen en el servidor

SSID de la sesión

Estas ideas son transparentes para el programador, no las tenemos que controlar, pero por higiene intelectual en la programación web, conv que

- Existen dos maneras de mantener el SSID de la sesión

1. Utilizando cookies, tema ya visto.
2. Propagando el SID en un parámetro de la URL. El SID se añade como una parte más de la URL, de la forma:

```
http://www.misitioweb.com/tienda/listado.php&PHPSESSID=34534fg4ffg34ty
```

- En el ejemplo anterior, el SID es el valor del parámetro PHPSESSID.

En php todas estas acciones se realizan de forma transparente para el programador, es decir, como desarrolladores podemos directamente u sesiones en php sin necesidad de tener que transmitir el SSID. Directemente **php** nos ofrece ***supervariables*** y ***funciones*** para gestionarlo

Configuración

Existen una serie de ***directivas*** para configurar las sesiones, que conviene conocer. Como toda configuración tendemos a mantener su estad lo cual es relativamente cómodo, pero no siempre práctico.

Ubicación del fichero de configuración

Dónde está el fichero de configuración de php

- Estas se pueden consultar viendo phpinfo(), y modificar en el fichero de configuración de php, ***php.ini***
- Para ver todas las directivas http://es.php.net/manual/es/session.configuration.php
- Algunas directivas de configuración que pueden resultar de interés

session.use_cookies

Indica si se deben usar cookies (1) o propagación en la URL (0) para almacenar el SID.
session.use_only_cookies
Se debe activar (1) cuando utilizas cookies para almacenar los SID, y además no quieres que se reconozcan los SID que se puedan pas de la URL (este método se puede usar para usurpar el identificador de otro usuario).

session.save_handler
Se utiliza para indicar a PHP cómo debe almacenar los datos de la sesión del usuario. Existen cuatro opciones: en ficheros (files), en m en una base de datos SQLite (sqlite) o utilizando para ello funciones que debe definir el programador (user). El valor por defecto (files) problemas en la mayoría de los casos.

session.name
Determina el nombre de la cookie que se utilizará para guardar el SID. Su valor por defecto es PHPSESSID.

session.auto_start
Su valor por defecto es 0, y en este caso deberás usar la función ***session_start()*** para gestionar el inicio de las sesiones. Si usas sesi web, puede ser buena idea cambiar su valor a 1 para que PHP active de forma automática el manejo de sesiones. No obstante por segu hacerlo de forma explícita cuando lo necesites

session.cookie_lifetime
Si utilizas la URL para propagar el SID, éste se perderá cuando cierres tu navegador. Sin embargo, si utilizas cookies, el SID se manter se destruya la cookie. En su valor por defecto (0), las cookies se destruyen cuando se cierra el navegador. Si quieres que se mantenga más tiempo, debes indicar en esta directiva ese tiempo en segundos.

session.gc_maxlifetime
Indica el tiempo en segundos que se debe mantener activa la sesión, aunque no haya ninguna actividad por parte del usuario. Su valor 1440. Es decir, pasados 24 minutos desde la última actividad por parte del usuario, se cierra su sesión automáticamente.

Creando la sesión

Qué es una de sesión

De forma colocial

Una sesión es la duración de la conexión que se establece entre un cliente (navegador/ip) y un servidor.

- Mientras dura la sesión puedo acceder a las variables de sesión establecidas
- Si abro un navegador y accedo a un sitio web, se establece la sesión
- Al cerrar el navegador se cierra la sesión esto es en teoría,

Tip: A veces se queda la sesión abierta por temas de cache o de la ***cookie*** que mantiene el id de sesión

- Por comparación es como si entrara en un centro comercial
- Mientras en estoy en él estoy dentro de mi sesión
- Cuando salgo se cierra la sesión
- Mientra estoy dentro podrían mantener información a usar en diferentes tiendas (productos comprados, pe.).

- En función de cómo esté configurado la directiva ***session.auto_start***
- Si esta activada, la sesión comienza automáticamente al conectarse a un sitio
- Si no está activada la iniciaremos con la funcion ***session_start()***;
- Un vez creada la sesión establecemos la variable y su valor en la superglobal $_SESSION

```
<?php
sesion_start();

$_SESSION['nombre']='manuel';
....

?>
*Ahora hemos establecido la variable de sesión '''''nombre'''''
*De esta forma en futuras navegaciones mientras dure la sesión o conversación entre el usuario cliente y el servidor podré acceder a la variable de sesion '''nombr
*En otro fichero php podré acceder al valor del nombre del  usuario
<?php
sesion_start();
...
$usuario = $_SESSION['nombre'];
....

?>
```

Tip: Una vez creada la sesión podemos almacenar/consultar información de la misma consultando la variable superglobal ***$_SESS***

Eliminando la sesión

- Se puede configurar para que de forma automática se eliminen los datos de una sesión pasados un determinado tiempo
- También podemos actuar directamente sobre una sesión eliminando información

session_unset.
Elimina las variables almacenadas en la sesión actual, pero no elimina la información de la sesión del dispositivo de almacenamiento

session_destroy.

Elimina completamente la información de la sesión del dispositivo de almacenamiento.

Tip: Cuando se hace un cambio de estado (login, cambio de permisos, ...): regenerar id.

```
session_regenerate_id()
```

Documentación

- http://www.php.net/manual/es/booSek.session.php
- http://www.w3schools.com/php/php_sessions.asp
- http://www.mclibre.org/consultar/php/lecciones/php_sesiones.html

Actividades

Contador de accesos a una página

- Haz un programa que cuente el número de visitas a una página en una misma sesión
- A la página podremos acceder por la url o haciendo un click en un botón de tipo submit

Contador de acceso a un usuario

- En este caso queremos contar cuantas veces accede un determinado usuario a nuestra página
- El programa visualizará en cada acceso el número total de accesos de cada usuario, indicando su nombre y número de ac

Programa una pequeña agenda

- Es una aplicación para mantener una pequeña agenda en una ***única*** página web programada en PHP.
- La agenda almacenará únicamente dos datos de cada persona: su nombre y un número de teléfono. Además, no podrá h nombres repetidos en la agenda.
- En la parte superior de la página web se mostrará el contenido de la agenda. En la parte inferior debe figurar un sencillo con dos cuadros de texto, uno para el nombre y otro para el número de teléfono.
- Cada vez que se envíe el formulario:

1. Si el nombre está vacío, se mostrará una advertencia.
2. Si el nombre que se introdujo no existe en la agenda, y el número de teléfono no está vacío, se añadirá a la agenda.
3. Si el nombre que se introdujo ya existe en la agenda y se indica un número de teléfono, se sustituirá el número de teléfo
4. Si el nombre que se introdujo ya existe en la agenda y no se indica número de teléfono, se eliminará de la agenda la entr correspondiente a ese nombre.

Control de acceso

- En este caso queremos mostrar un formulario con nombre y password ("alumno", "password")
- Dejamos solamente 3 intentos seguidos para acceder

- En caso de el intento sea incorrecto, mostraremos un mensaje "Datos incorrectos, le quedan X intentos"
- En caso de que insertemos correctamente iremos a una página llamado sitio.php donde mostraremos un mensaje de bier nombre de la persona que accede
- Será posible navegar a una segunda página llamada productos.php donde se mostrará otro mensaje con el nombre del u
- Desde la segunda página se podrá cerrar sesión, en cuyo caso deberemos de pasar a la primer página
- Estas dos páginas solo deberá de ser posible acceder si nos hemos logueado, si no, no nos dejará, mostrando un mensaj reenviándonos a la página index.php en 3 segundos.

Css para el login

Usuario:ManuelRomero/ProgramacionWeb/php/Aut Ses Coo/Cookies

De WikiEducator
< Usuario:ManuelRomero | ProgramacionWeb/php

BLOQUE 2 PHP: **PROGRAMACIÓN ORIENTADO A OBJETOS**

¡Construyendo componentes!

PHP Como lenguaje orientado a objetos

Autentificación | Sesiones | **Cookies** | Ejercicios | Práctica | Volver

cpiip Los Enlaces

TEMA 7.3 COOKIES

Contenido

Qué es una Cookie

Una cookie es una pequeña cantidad de datos almacenada por el navegador del usuario cuando solicita una página a un servidor. Básicamente consiste va a ser cantidad de información que almacena preferencias de un usuario. En parte se podría pensar que con esta técnica podemos suplir la característica de que http es un protocolo sin estado (Insistir en que esto es una característica del protocolo, no una limitación). Una ***Cookie'*** *es un fichero que se* **almacena en el cliente** *y guarda información de ese cliente en referencia a una determinada sitio web.*

Tip: Como se guardan en el cliente, se necesita confirmación expresa del cliente para dejar que se almacenen cosas en su equipo

Legislación sobre las *cookies*

- Es muy importante respetar la legislación en el uso de cookies
- Básicamente consiste en que hay que avisar al cliente de que se están utilizando
- En cualquier caso conviene estar al día.

http://www.agpd.es/portalwebAGPD/canaldocumentacion/publicaciones/common/Guias/Guia_Cookies.pdf

- En general no se debe de abusar de ellas; hay que tener en cuenta que en muchos las pueden tener deshabilitadas y esto puede impedir que nuestra aplicación web se ejecute de forma eficiente o incluso correcta. Por ello no debería de ocurrir que una aplicación web dependiera de los valores de cookies para funcionar de forma correcta

Puntos a tener claros

El servidor es quien solicita la creación del coockie en el cliente
El cliente crea un fichero con dicha información
Una vez creada, solo puede ser leída por el sitio web que la creó.
Para trabajar con cookies tenemos que hacer dos cosas

1. Almacenar o crear la cookie
2. Poder leerla en un momento determinado

- El tamaño máximo de un cookie está limitado a 4K.

Crear una coockie

- Para ello usaremos la función ***setCookie(...)***
- En esta función puede recibir hasta 7 parámetros, (sólo el primero es obligatorio).

```
http://es.php.net/manual/es/function.setcookie.php
```

- Es habitual usar los tres primeros que serían

1. Nombre de la cookie
2. Valor que almacenamos en ella
3. Tiempo de expiración: es un entero en segundos. Si no se especifica, la cookie termiará junto a esta sesión.

- Por ejemplo si quieremos almacenar el usuario y que tenga un tiempo de duración de 1 hora

```
setcookie("usuario", $_SERVER['PHP_AUTH_USER'], time()+3600);
```

- Importante el envío de cookies al cliente ha de hacerse antes de que se escriba nada en html, igual que las funciones header.

Recuperar la cookie

- El proceso de recuperación de la información que almacena una cookie es muy simple.
- Cuando accedes a un sitio web, el navegador (cliente), le envía de forma automática todo el contenido de las cookies que almacene relativas a ese sitio en concreto (servidor).
- Desde PHP se puede acceder a esta información por medio del array $_COOKIE.
- Para recuperar los datos anteriores

```
$usuario = $_COOKIE['usuario'];
```

Borrar una cookie

- Para borrar la cookie usamos la función setcookie con un tiempo negativo

Tip: Tener en cuenta que el tiempo es respecto al segundo o UNIX

```
<?php
// Ponemos un tiempo de expiración negativo
setcookie("user", "", time()-3600);
?>
```

Información sobre visitas

Cookies con información de visitas

Resumen

Creación de coockies

```
<?php
$expire = time()*60*60*24*30 //La coockie durará un  mes
setcookie("user", "Alex Porter", $expire);
?>
```

- Si tiempo de expiración es 0, la coockie que solo dura la sesion

```
<?php
$expire = =; //La coockie durará lo que dure la sesión
setcookie("user", "Alex Porter", $expire);
?>
```

Ver el valor de una cookie

```php
<?php
// Ver una cookie concreta
echo $_COOKIE["user"];

// Para ver todas las cookies
print_r($_COOKIE);
?>
```

Borrado

```php
<?php
// Ponemos un tiempo negativo
setcookie("user", "", time()-3600);
?>
```

Coockies con array

- En este caso la cookie en lugar de contener un solo valor va a contener un array
- A la hora de leer la cookie se hace igual que hasta ahora
- Supongamos que tenemos varias cookies, dependiendo del usuario que se ha conectado
- De cada uno de esos usuarios tendríamos los accesos que ha realizado en un array indexado indicando la hora
- Leemos la cookie

```php
<?php
....
$usuario = $_POST['usuario']
......
//Leemos la cookie que es un array de accesos
$arrayAccesos = $_COOKIE[$usuario];
.....
```

- Guardar la cookie
- Esto crea el plateamiento de como guardar la cookie a la hora de qué nombre va a tener esta cookie
- Es como que en la misma cookie queremos guardar muchos valores, pero en realidad guardamos uno solo que es el array
- Una manera de hacerlo es incorporar a la cookie ya guardada el nuevo índice
- Para ello una manera de hacerlo es obtener el valor del indice del nuevo elemento obteniendo el tamaño del vector hasta ahora
- Incorporamos en la posición correspondiente el nuevo valor

```php
....
$usuario = $_POST['usuario']
......
//Leemos la cookie que es un array de accesos
$arrayAccesos = $_COOKIE[$usuario];
.....
//Ahora ya puedo obtener el tamaño del vector
$indice = count($arrayAccesos);
//Con este valor añado la nueva entrada
setCookie($usuario.'['.$indice.']',time(),time()+3600
.....
```

Administrar cookies en diferentes navegadores

- ***Firefox***: http://support.mozilla.org/es/kb/Borrar%20cookies?esab=a&s=cookies&r=2&as=s

- ***Chrome/Chromium***: http://support.google.com/chrome/bin/answer.py?hl=es&answer=95647
- ***Internet Explorer***: http://windows.microsoft.com/es-es/windows7/how-to-manage-cookies-in-internet-explorer-9
- ***Opera***: http://help.opera.com/Windows/11.50/es-ES/cookies.html
- ***Safari***: http://support.apple.com/kb/ph5042

Referencias en la web

- http://docs.php.net/manual/es/features.cookies.php
- http://docs.php.net/manual/es/function.setcookie.php
- http://www.w3schools.com/php/php_cookies.asp
- http://www.mclibre.org/consultar/php/lecciones/php_cookies.html

Normativa: Directiva 2009/136/CE

Usuario:ManuelRomero/php/NewPHP/B2T8/BD

De WikiEducator
< Usuario:ManuelRomero

Bases de datos: El servidor es fundamental

¡Las bases de datos: Parte fundamental en el servidor de desarrollos web

PHP Un lenguaje de script al lado del servidor

Bases de datos en PHP | Ejercicios | Práctica | Volver

TEMA 3: LENGUAJE PHP

Show presentation

Contenido

BASES DE DATOS: Introducción

Una muy breve introducción sobre lo que es una ***base de datos***. Es éste un concepto conocido, pues se ha debido de estudiar previamente.
En cualquier caso vamos a aclarar a nivel intuitivo conceptos importantes para usar posteriormente.

Base de datos

Una base de datos es una colección o conjunto de datos que vamos a almacenar en un dispositivo de almacenamiento permanente (generalmente HD), que tiene una determinada estructura u organización, la cual nos va a permitir operar de una forma organizada con esos datos

Sería inimaginable buscar un libro en una biblioteca si no hubiera una organización u orden para localizarlo o a la hora de añadir un libro nuevo (en una sección, en una estantería concreta y no en cualquiera.

Igualmente si voy a tener libros pequeños, los pondré en estanterías pequeñas, si voy a almacenar libros grandes necesitaré tener estanterías grandes.
Siguiendo esta lógica, las bases de datos han de estar preparadas para almacenar el tipo de información que nos pueda venir, para ello habrá que hacer un diseño correcto de las tablas y atributos para poder almacenar toda la información de nuestro sistema.

Puntos clave

El diseño de la base de datos es un factor fundamental para el éxito de la aplicación

Puntos clave

El diseño de la base de datos se debería de hacer solo una vez y modificarlo pocas veces durante su vida

Normalización

Las bases de datos han de estar construidas de una forma ***normal***, de manera que evitemos redundancias innecesarias, y sólo las mantengamos cuando las consideremos necesarias y seamos conscientes de que existen.

- Si hacemos nuestros diseños usando el modelo de chen, garantizamos hasta la ***3FN***

Resumen

1FN
Toda tabla tiene una clave primaria y no puede haber valores **multivaluados** en la misma tupla

2FN
Todo atributo no primo depende de forma completa de la clave y no solo de parte de ella

3FN
Los atributos no primos no implican a otros atributos no primos

Forma normal de Boyce y Codd.

Los atributos primos que forman una clave no se implican entre ellos

Tip: Atributos ***primos*** son los que forman parte de alguna **clave candidata**

Tip: Atributos ***no primos*** son los que no forman parte de ninguna **clave candidata**

Usar BD desde un lenguaje de programación

Una de las principales características que tiene la programación del ***back-end*** o programación al lado del servidor es acceder a bases de datos. Esta parte sí que es intrínseca y propia del lado del servidor. Nosotros accederemos a la base de datos desde nuestro programa. Desde él de manera habitual, realizaremos las siguientes acciones :

Conectarnos a la base de datos
: Para ello necesitamos un software específico del gestor de bases de datos con el que vayamos a trabajar.

Seleccionar
: La base de datos con la que vamos a trabajar.

Trabajar con Bases de datos

Actuar
: acciones a hacer con la base de datos
: son las habituales (consultas, inserciones, modificaciones y/o borrados)

Procesar información
: En caso de consultas deberemos recorrer el cursor u objeto que nos retorne la consulta
: Siempre contendrá el conjunto de filas devueltas (en caso de que haya).

Cerrar la base de datos
: Es importante no dejar conexiones abiertas de forma innecesaria

Para realizar estas acciones disponemos de diversas ***Funciones/Clases*** específicas dentro de PHP, Nos referiremos a ellos como ***extensiones*** de PHP.

Puntos clave

Independizar la base de datos y el lenguaje de programación

concepto de driver, conector y extensión (mysql, mysqli, PDO).

Lectura recomendada

```
http://php.net/manual/es/mysqli.overview.php
http://php.net/manual/es/book.pdo.php
http://php.net/manual/es/book.mysqli.php
```

Bases de datos y PHP

PHP tiene un ***API*** especifico para trabajar directamente con mysql ***mysqli***, el cual incorpora el driver y conector necesario para trabajar con ella de forma nativa. Que el driver sea nativo implica que está implementado utilizando un **framework** de extensiones de php.
También vamos a disponer de la extensión PDO, la cual se independiza del gestor concreto de bases datos que vayamos a utilizar.

Extensiones de php

- Por lo tanto en este tema vamos a ver dos extensiones:

1. ***mysqli*** usar una extensión nativa con su SGBD en concreto mysql que viene con el propio lenguaje
2. ***PDO*** usar una extensión genérica que permite conectarse con cualquier gestor de BD, sin necesidad de cambiar nada de código, salvo los parámetros de la construcción del objeto.

Sistema Gestor de Bases de datos

Sistema Gestor de Bases de Datos (SGBD)

Un SGBD es un ***software*** que nos va a permitir ***gestionar*** una base de datos.

Gestionar una Base de datos

Consiste de crear la base de datos y manipularla. manipular implica poder realizar operaciones sobre ella como son

Acciones sobre la base de datos

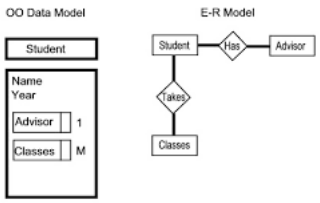
Sobre los elementos de la base de datos (tablas, usuarios, indices, ...

- Crear
- Eliminar
- Modificar

Sobre el contenidos de los elementos (En la tablas)

- Insertar
- Actualizar
- Modificar o cambiar valores
- Consultar

La manera en la que se organiza la información para que se pueda gestionar es fundamental, y los software de los gestores de bases de datos han de gestionarla siguiendo ese criterio.
De esta forma aparece una nueva letra, así hablamos de SGBD***R***, haciendo referencia a que son Relacionales, o SGBD***OO*** haciendo referencia que son orientados a objetos.



Es éste un tema que lleva muchos años en uso.Para más información:

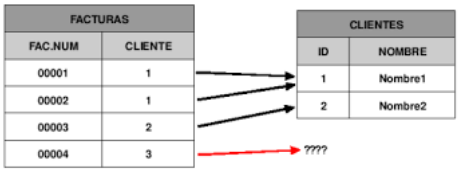
Recursos de la Web

- http://di002.edv.uniovi.es/~labra/cursos/ver06/pres/XMLBD.pdf
- http://www.cs.us.es/blogs/bd2012/files/2012/09/BD-Tema-5.pdf

Los Sistemas de Bases de datos Relacionales están basados en **Tablas** y en **claves principales - claves foráneas** para mantener relacionadas **tuplas** o filas de diferentes tableas.

Imagen Cliente Factura

- En la imagen siguiente vemos una tabla de clientes y una facturas
- Vemos cómo cada factura corresponde a un cliente
- Para ello en la tabla de factura tenemos la clave de cliente
- No tiene sentido que en ese campo haya un identificador de un cliente que no existe
- Esta restricción es una **restricción de clave foránea** o ***restricción de integridad referencial***



- Entre los sistemas de bases de datos Relacionales tenemos ***Mysql*** Vs ***maria***, ***PostgreSQL'*** *(Aunque ésta última es Objeto Relacional),* **Oracle** *que también tiene la posibilidad de ser usada como objeto-Relacional, ...*

Gestores relacionales más utilizados

- http://revistadigital.inesem.es/nuevas-tecnologias/los-gestores-de-bases-de-datos-

mas-usados/

Actualmente también se trabaja de forma complementaria en las aplicaciones con SGBD NoSql, llamados así por que su estructura no está basada en relaciones entre tablas, sino que tiene otros criterios.
Podemos ver que existen bases de datos que siguen otro criterio, como el de almacenar con clave -valor como hace ***MongoDB*** que está basada en ***Documentos*** , por lo que se llama una base de datos **Documental**. Este concepto lo estudiaremos en otro tema. Para gestionar sistemas de bases de tipo Mongo, podemos usar herramientas como ***Rockmongo***, o ***Mongochef***. Esta última es más recomendada .

Recursos de la Web

- ***RockMongo*** http://mongodb-tools.com/tool/rockmongo/
- ***Mongochef*** https://studio3t.com/

Bases de datos no Sql

http://www.campusmvp.es/recursos/post/Fundamentos-de-bases-de-datos-NoSQL-MongoDB.aspx

SQL

SQL es un lenguaje de consultas estructuradas o Structured Query Language. Es un lenguaje de 4º generación donde el programador o usuario del lenguaje especifica lo ***qué*** quiere, pero no establece el **procedimiento** a seguir para conseguirlo, no dice ***cómo*** ha de hacer el sistema para conseguirlo

Lenguaje no procedural Vs lenguaje procedural)

Con lenguaje no procedural (tipo SQL)

```
Dime todos los empleados de más de 45 años
o
SELECT *
FROM empleados
WHERE edad>45;
```

Lenguaje procedural (tipo php o java)

```
Abre la tabla empelados
Lee desde el primer registro hasta el último (Bucle)
     Verifica si ese empleado tiene mas de 45
     Si es así añádelo a la lista de resultado
Muestra la lista de resultado
```

SQL

Lenguaje para gestionar bases de datos **relacionales** cuyas ***instrucciones o sentencias*** especifican lo ***qué*** se quiere, no ***cómo*** se va a realizar

Lenguajes dentro de SQL

SQL no es, como su nombre indica, solamente un lenguaje de consultas, sino que proporciona todas las instrucciones necesarias para ***gestionar*** una base de datos.
SQL incorpora tres tipos de lenguajes: de ***Definición***, *de* **Control**, *de* **Manipulación**.
Las instrucciones de ***SQL*** se conocen como ***sentencias***, y todas se caracterizan por que empiezan por una palabra reservada que identifica a qué lenguaje de los tres que tiene ***SQL*** pertenece esa instrucción.

Lenguajes SQL

DDL

Lenguaje de ***definición*** de datos.

Las sentencias de este lenguajes permite crear nuevos elementos en la base de datos como usuarios, tablas, índices, etc.

DCL

Lenguaje de ***control*** de datos

Las sentencias de este lenguaje permite gestionar permisos sobre los elementos y acciones sobre la consistencia de los datos

DML

Lenguaje de ***manipulación*** de datos

Sin duda son las instrucciones o sentencias mas utilizadas en un Gestor
Con ellas podemos manipular lo que se conoce como la extensión o contenido de la base de datos.
Es decir actuar sobre las tuplas (insertar, modificar y borrar), y consultar

Sentencias DDL

Sentencias DDL

CREATE

```
CREATE DATABASE usuarios;
CREATE TABLE usuario(
       password VARCHAR(200),
       nombre varchar2(40)
);
```

DROP

```
DROP TABLE usuario;
```

ALTER

```
ALTER TABLE usuarios ADD direccion VARCHAR(20);
```

Sentencias DCL

Sentencias DCL

GRANT

para asignar privilegios

```
GRANT ALL ON DATABASE.* TO 'manuel'@'localhost'
#Asigna todos los privilegios sobre las tablas
#de la base de datos 'database'
# al usuario 'manuel'
```

REVOKE

Elimina privilegios asignados

COMMIT

Confirma una transacción de manera permanente en la base de datos

ROLLBACK

Desace todos los cambios en la base de datos desde la ultima transacción confirmada

Sentencias DML

Sentencias DML

INSERT

Inserta tuplas en una tabla
Esta acción pude devolver algún tipo de valor booleano indicando se se ha insertado o no la tupla

```
INSERT INTO usuarios (nombre, password)
VALUES("manuel", "password_no_segura");
```

DELETE

Borra tuplas (0 o más) de una o más tablas

```
DELETE FROM usuarios WHERE nombre="manuel";
```

UPDATE

Actuliza tuplas (0 o más) de una o más tablas
Las dos últimas sentencias (***UPDATE Y DELETE***) suelen devolver un entero.
Este valor indica el número de tuplas actualizadas o borradas (0 o n)
Un error se detalla con el valor (-1 o false).

```
UPDATE usuarios SET nombre="alicia" WHERE nombre="manolo";
```

SELECT

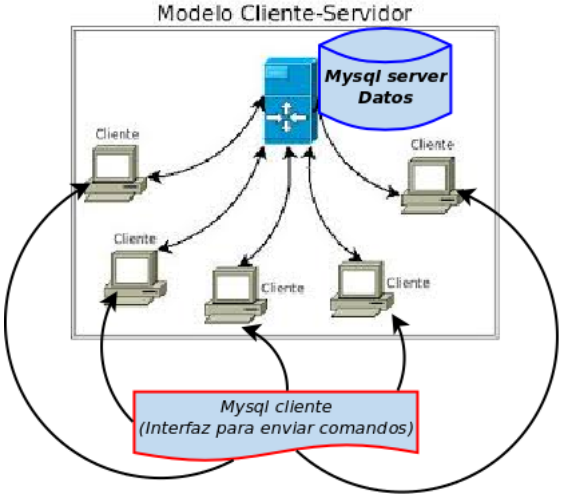
Es esta la sentencia estrella. Usando el álgebra relacional, permite recuperar un subconjunto de los datos de la base de datos. Esta sentencia devuelve un conjunto de tuplas devueltas por la consulta. Normalmente va a devolverlo como un objeto dependiendo del lenguaje. Dentro de esta **sentencia**, se pueden encontrar 5 **cláusulas** de las cuales solo las dos primeras son obligatorias. Simplemente por recordarlas, ya que son contenido del módulo de primero de **Bases de datos**.

```
  SELECT
  FROM
  WHERE
  GROUP BY
   HAVING
(ORDER BY)
```

- ***Having*** solo se usa como opción en el **group by**
- ***Order by*** es un criterio de ordenación y no se considera como cláusula propiamente dicho, por eso la he puesto entre paréntesis.

Mysql

Mysql es un gestor de bases de datos relacional. Actualmente es un producto propietario de la empresa ***Oracle***, y el software correspondiente libre es ***Maria*** que en estos momentos funciona exactamente igual que ***Mysql*** y con los mismos comandos. (año 2017).
Tener en cuenta la diferencia de la parte del servidor (donde realmente se guardan los datos), y la parte del cliente que es un software que nos ofrece una interfaz de comandos o gráfica y nos permite conectarnos al servidor y gestionar la bases de datos (básicamente enviar sentencias SQL y recoger los resultado).




```
apt-get install mysql-server mysql-client
```

Como todos servicio se puede arrancar o parar

```
service mysql start/stop/restart/status
```

- El fichero de configuración

```
/etc/mysql/my.cnf
```

Mysql

- En él podemos ver el puerto (3306 por defecto), el usuario y otros parámetros del servicio

Mysql con docker

- Monta un contenedor de docker llamado contenedorMysql.
- Instala en el contenedor mysql.
- Prueba desde tu localhost a conectarte a la base de datos de ese contenedor.

Posible solución [▼]

Mysql desde línea de comandos

Es importante saber manejar la línea de comandos con el cliente **mysql**. En alguna ocasión puede ser la forma más rápida de obtener información o realizar acciones sobre la base de dados. Para ello debemos de repasar algún comandos. Son pocos, pero hay que conocerlos.

Acciones básicas

1. Conectarse a una base de datos en un servidor

```
mysql -u usuario -ppassword
# Si no especificas el passowr, se solicitará
# Si lo especificas ha de ir sin ningún espacio entre el  parámetro -p
```

1.- Mostrar las bases de datos

```
SHOW DATABASES;
```

2.-Usar una base de datos concreta

```
USE nombre_base_datos;
```

3.-Mostrar la descripción de una base de datos (Las tablas que contiene)

```
DESCRIBE nombre_tabla;
```

4.-Obtener ayuda de los comandos disponibles

```
help;
```

- Por supuesto debemos de poder ejecutar sentencias SQL

5.-Crear una base de datos (Sentencia SQL), y borrarla 6.-Crear una tabla 7.-Crear un usuario y darle permisos 8.-Modificar y borrar una tabla 9.-Insertar, modificar y borrar registros o tuplas 10.-Realizar consultas ***select***

```
connect -h,-u, -p
 use database
 exit o quit
```

```
help Para conocer los comandos que se pueden usar.
     http://ftp.nchu.edu.tw/MySQL/doc/refman/5.0/es/sql-syntax.html
```

Actividad

1. Conectar a mysql
2. Mira Las bases de datos que tienes
3. usa una base de datos concreta
4. mira las tablas que tiene esa bases de datos
5. mira la estructura de la tabla
6. Haz una consulta de los valores que tiene su primera columna
7. ejecuta el comando help
8. sal de sql

Bases de datos

- Vamos a trabajar con un ejemplo de bases de datos de una tienda

Archivo:BaseDatos.pdf

Click para ver el contendio [▼]

Recordando mysql

- En la siguiente página puedes hacer un repaso de mysql

```
http://dev.mysql.com/doc/refman/5.7/en/index.html
```

- Un manual en castellano

```
http://ftp.nchu.edu.tw/MySQL/doc/refman/5.0/es/
```

- Debemos conocer también la herramienta phpmyadmin

```
sudo apt-get install phpmyadmin
```

Recordando mysql en la página oficial

```
http://www.phpmyadmin.net/home_page/index.php
```

Mysql Interfaz gráfica

Además de línea de comandos también se dispone de varias herramientas gráficas alternativas. Siempre resulta más agradable e intuitivo, pero requiere más recursos y tener instalados aplicaciones concretas.
En windows o con **wine** en linux podemos probar ***SQLyog*** https://www.webyog.com/product/sqlyog, una herramienta intuitiva y de poco peso. No obstante usaremos ***phpmyadmin***, por ser un clásico y de sencillo uso.

Actividad

- Instalamos phpmyadmin en nuestro contenedor contenedorMysql

```
sudo apt-get install phpmyadmin
```

- Probamos a acceder a él desde nuestro localhost
- Depende de como instalemos es posible que haya que hacer un enlace simbóloco.
- Para ello hay que buscar dónde se ha instalado el ***index.php*** de ***phpmyadmin*** .
- Entonces haceremos un enlace simbólico de esa carpeta a nuestro DocumentRoot.
- En mi caso:

```
# ln -s /usr/share/phpmyadmin /var/www/phpmyadmin
```

Probamos a conectarnos por phpmyadmin a nuestro servidor local

```
#Escribimos en el url
http://ip_contenedor/phpmyadmin
```

- Para la base de datos anterior, vamos a establecer un poblar la BD.
- A continuación el contenido de un sql para poblar la BD

Click para ver el contenido [▼]

Actividad

Usando phpmyadmin, carga los datos para poblar la base de datos

- Otra herramienta importante que permite realizar diseños es workbrench

```
http://dev.mysql.com/doc/workbench/en/index.html
```

```
sudo add-apt-repository ppa:olivier-berten/misc
sudo apt-get update
sudo apt-get install  mysql-workbench
```

Actividad

- Instalamos y probamos las diferentes opciones con esta herramienta
- Probamos a hacer un diseño en modelo relacional y a partir del modelo creamos las tablas.
- Probamos las opciones ***forward engineer*** y ***reverser engineer***

Herramientas de administración

- Con ***phpmyadmin***, podemos hacer casi todo de administrar y manejar las bases de datos
- No obstante mysql proporciona una serie de herramientas que permiten administrar por línea de comandos .
- En muchas ocasiones este tipo de operaciones resultan muy interesantes.

1. mysql
2. myhsqladmin
3. mysqlshow

mysqladmin

- Es un cliente específico para la administración
- Entre otras acciones podemos realizar:

1. crear y eliminar bases de datos.
2. mostrar la configuración y el estado del servidor.
3. cambiar contraseñas.
4. detener un servidor.
5. ver la version del servidor

```
http://ftp.nchu.edu.tw/MySQL/doc/refman/5.0/es/mysqladmin.html
```

Actividad

Prueba a hacer cada una de las acciones especificadas anteriormente

```
mysqladmin -u root -proot create nombreBaseDatos
mysqladmin -u root -proot extend-status
mysqladmin -u root -proot password
mysqladmin -u root -proot shutdown
mysqladmin -u root -proot version
```

mysqlshow

```
http://ftp.nchu.edu.tw/MySQL//doc/refman/5.0/es/mysqlshow.html
```

- muestra información sobre la base de datos

```
mysqlshow -u root -proot
```

- Nos mostraría las tablas de ese usuario

Mysqli

Para trabajar con las extensiones, las usaremos siempre orientadas a objetos, aunque tengan la correspondiente funcionalidad en el lenguaje estructurado.

- Para recordar muy brevemente posemos usar el siguiente enlace

```
http://www.desarrolloweb.com/articulos/1540.php
```

Uso básico de POO

- Recordamos que para crear una nueva instancia de una clase usamos el operador ***new***

```
$miObjeto = new Clase();
```

- Para acceder a los diferentes métodos del objeto instaciado, usamos el operador de indirección **->**

```
$miObjeto->metodo($parametros);
```

Mysql y su extensión mysqli para php

CONECTARNOS A LA BASE DE DATOS

- A continuación iremos viendo como implementar las acciones básicas en el lenguaje

Conectarse

- Para conectarse a una base de datos , creamos una instacia de la clase mysqli de la forma

```
$miConexion = new mysqli(....);
```

Extensión Mysqli

- El constructor de la clase puede recibir hasta 5 parámetros, de los cuales 4 se suelen usar con bastante frecuencia

1. ***$host*** nombre o ip del equipo (null o localhost, identificaría el equipo actual).i
2. ***$usario*** es el usuario de la base de datos
3. ***$pass***
4. ***$nombreBD***
5. ***$puerto***
6. ***$shocket***

Ejemplo new mysqli(...)

```
$host="localhost"
$usuario="manolo";
$pass="romero";
$nombreBD="alumnos";

$miConexion = new mysqli ($host,$usuario,$pass,$nombreBD);
if ($miConexion==null)
    echo"No se ha podido crear el objeto";
else
    echo "Objeto creado";
```

mysqli(...)

- Esta función retorna el recurso de la conexión.

Tip: Aunque hablemos de mysqli como una clase, en realidad es una clase especial conocida como recurso.

Un recurso a diferencia de una clase no se puede serializar para pasar entre scripts.

- Para gestionar los errores debemos de usar el atributo ***connect_error*** de la clase ***mysqli***.
- El echo de que se pueda instanciar o el objeto de la clase, no implica que se haya realizado la conexión.
- Este atributo aporta información sobre el error o contiene null si no se ha producido ninguno.
- En el código anterior

```
if ($miConexion->connect_error)
  echo "Error conectando con la base de datos: ".$miConexion->connect_error;
```

- Para ver información sobre la conexión se puede usar los atributos ***$server_info*** o ***$host_info***

Resumen

```
$conesion= new mysqli($host,$user,$pass,$bd);
$conesion->conect_error
$conesion->conect_errno
$conesion->server_info
$conesion->host_info
```

Opciones por defecto

- Hay muchas opciones de **mysqli** que se pueden configurar en el fichero ***php.ini***, no es algo que normalmente se modifique, pero por curiosidad las comentamos aquí.
- Aquí tenemos alguna de ellas

mysqli.allow_persistent
Permite crear conexiones persistentes.
mysqli.default_port
Número de puerto TCP predeterminado a utilizar cuando se conecta al Servidor de base de datos.

mysqli.reconnect
Indica si se debe volver a conectar automáticamente en caso de que se pierda la conexión.
mysqli.default_host.
Host predeterminado a usar cuando se conecta al servidor de base de datos.
mysqli.default_user.
Nombre de usuario predeterminado a usar cuando se conecta al servidor de base de datos.
mysqli.default_pw
Contraseña predeterminada a usar cuando se conecta al servidor de base de datos.

- La lista completa la podemos ver en el siguiente link

```
http://php.net/manual/es/mysqli.configuration.php
```

Actividad

- Configura dicho fichero, para poder conectar a la base de datos sin aportar parámetros al constructor
- Luego déjalo como estaba :)

Cambiar la base de datos

- Si hemos seleccionado una base de datos, o no hemos seleccionado ninguna y queremos cambiar a otra

```
$miConexion->select_db("nombre_base_datos");
```

- Cuando ya no vamos a usar un recurso, conviente y repito CONVIENE, liberarlo.

```
$miConexion->close();
```

Tip: Es muy importante liberar un recurso reservado una vez que ya no se vaya a usar mas.

DML

- En SQL sabemos que tenmos tres tipos de lenguajes DDL, DML, DCL
- Ahora toca DML, Leguane de maninputación de datos
- Podemos clasificar en dos tipos de clúsulas:

1. las que no devuelven registros de datos (INSERT, DELETE, UPDATE)
    1. Generalmente retornan un entero que es el número de filas aceptadas o un booleano que indica si se realizó no la operación

DML

1. Las que pueden retornan una colección de filas (SELECT), generalmente conocidas como cursor.

- En mysqli podemos enviar cualquiera de estas claúsulas con el método ***query***

INSERT, UPDATE Y DELETE: Método *query(...)*

El método ***query***, es un método de la clase ***mysqli*** que permite enviar una **sentencia sql** a la base de datos con la que tengamos conexión. En función del tipo de consulta, el método nos puede devolver los siguientes valores:

1. ***Booleano*** (con las sentencias **Insert, Update, Delete**); indica si la acción se realizó o no correctamente.
2. ***mysqli_result*** (con la sentencia **Select**); si la consulta es de tipo ***select***, el método retornará un conjunto de filas (0

o más); Dispondremos de esta información en un objeto del la clase **mysql_result**. Esta clase la estudiamos posteriormente.

Hay que tener en cuenta que este método modifica el objeto de la clase ***mysqli*** pudiendo afectar a los siguiente atributos:

Atributos que puede modificar *query()*

1. ***$affected_rows***. Número de filas modificadas por la acción.
2. ***$error, $errno***. En caso de producir un error la sentencia.
3. ***$insert_id*** Devuelve el id autogenerado que se utilizó en la última inserción.

Resumen

```php
//Me conecto a la base de datos
$miConexion = new mysqli("localhost", "root", "root", "dwes");

//Me preparo sentencias
//Actuaré sobre la tabla tienda (ver BD anterior)
//Tiene los campos cod (autogenerado de forma incremental),
//nombre y tlf (ambos de tipo varchar)
$sentenciaInsert="INSERT INTO tienda (nombre, tlf)
  VALUES ('Tienda centro', '111-155226')";
$sentenciaDelete="DELETE FROM tienda
  WHERE nombre = 'Tienda centro' ";
$sentenciaUpdate="UPDATE tabla tienda
 SET nombre = 'Tienda principal'
 WHERE  nombre = 'Tienda centro'";


//Hago una consulta de tipo insert
$resultado=miConexion->query($sentenciaInsert)
if($resultado){
   echo"Se han insertado  $miConexion->affected_rows
        filas en esta acción <br />";
   echo "en la inserción se asignó el id autogenerado $miConexion->insert_id
        <br />";
}
```

Tip: *Muy importante* observa el tema de las comillas, los valores de las instrucciones sql sin son de tipo varchar deben de ir entre comillas

'*Tip:* Muy importante ***para la BD es igual comilla simple '''''*** que comillas dobles ***"***

Pregunta

Qué pasa si en nombre de la tienda es por ejemplo Technology's house

- Habría que escapar ese caracter

Escapar caracteres

Consiste en que de una forma automática si una cadena de caracteres tiene comillas, que estás queden escapadas para que formen parte de la cadena de caracteres y no especifique delimitación de la cadena.

Método para escapar cadenas

- Es un método de la clase ***mysqli***

```
string mysqli_real_scape_string ( mysqli $conexion, string $cadena_a_escapar )
```

También se puede usar como método de la clase

```
$conexion = new mysqli(..);
$cadena = $conexion->real_scape_string(....);
```

Escapar caracteres

- suponemos que tenemos una tabla llamada **acciones** que tiene el campo **descripcion** y **cod_usuario**
- Quiero insertar este valor para descripcion como muestra el siguiente código:

```
//...
$descripción  = "I don't want go animal's house"
//Esta asignación es correcta
//...
$consulta = "INSERT INTO acciones VALUES (1, '$descripcion')"
//...
```

si yo miro el contenido de la variable ***$consulta*** que es lo que pasare al método ***query()***

```
INSERT INTO acciones VALUES (1, 'I don't want go animal's house')
```

- Esto a la hora de insertar me generará un error, ya que el gestor interpreta que el contenido para el campo ***descripcion*** es 'I don'

- Para evitar esto yo querría que el valor de ***$descripcion*** estuviera escapado

```
INSERT INTO acciones VALUES (1, 'I don\'t want go animal\'s house')
```

```
se puede conseguir
```

```
//...
$descripción  = mysqli_real_scape_string("I don't want go animal's house");
//...
$consulta = "INSERT INTO acciones VALUES (1, '$descripcion')"
//...
```

Tip: Observa el tema de las comillas, los valores de las instrucciones **sql** sin son de tipo ***varchar*** deben de ir entre comillas

```
$miConexion->query($sentenciasDelete);
$resultado =$miConexion->query($sentenciaDELETE);
if ($resultado){
    echo "Se han borrado $miConexion->affected_rows filas ";
}

$miConexion->query($sentenciasUpdate);
$resultado = $miConexion->query($sentenciaUPDATE);
if ($resultado){
    echo "Se han actualizado $miConexion->affected_rows filas ";
}
</sourece>
</div>
<div class="slide">

<source lang=php>
$miConexion->query($sentenciasInsert);
$resultado = $miConexion->query($sentenciaINSERT);
if ($resultado){
    echo "Se han insertado $miConexion->affected_rows filas ";
}
```

- Observemos su uso en el ejemplo

```
  //Establecemos la conexión
  $miConexion = new mysqli('localhost', 'manolo', '12345.', 'baseDatosPrueba');

  //Capturamos un posible error
  $error = $miConexion->connect_errno;

  //En caso de error informamos de ello
  if ($error == null) {
    $resultado = $miConexion->query('DELETE FROM stock WHERE unidades=0');
     if ($resultado) {
        print "<p>Se han borrado $miConexion->affected_rows registros.</p>";
    }
  }
  $miConexion->close();
}
```

Clausula SELECT con query

- Tenemos dos maneras de realizar consultas con mysqli

1. query
2. real_query

- En el primero caso el método nos retorna un cursor que será de la clase ***mysqli_result***.
- En el segundo caso nos retornará un booleano y para leer los datos deberemos usar o ***store_result*** o ***use_result*** según veamos a continuación.

Método query

- Una vez que tenemos los datos almacenados debemos saber acceder.
- Tenemos 4 formas de poder acceder a los datos según usemos un método u otro.
- Cuando hablamos de acceder a los datos, estamos estableciendo la forma de extraer cada fila, registro o tupla resultado de ejecutar la consulta.

fetch_array()
Va obteniendo cada registro como un array
Este array podemos usar tanto de forma indexada, como asociativa (con el nombre del campo)

fetch_assoc()
En este caso el array que retorna es asociativo

fetch_row()
En este caso el array que retorna es indexado

fetch_object()
En este caso en lugar de retornar un array, retorna un objeto, donde cada campo son los diferentes atributos de ese objeto

- En todos los casos cada vez que leemos un elemento de mysqli_result, lo que por comparativa sería un cursor, vamos avanzando al siguiente. Cuando hayamos leído todos retornaría null

Actividad

- Obtén todos los registros de la tabla familia
- visualízalos en una tabla usando los tres modos de lectura de datos vistos anteriormente

- Para liberar un recurso del tipo ***mysqli_result***, usamos el método ***free()***;

- La clase ***mysqli_result***, además de los métodos vistos tiene un par de atributos interesantes

int $field_count;

Nos dice cuantas columnos tiene el query actual

int $num_rows;

Nos dice cuantas filas hemos obtenido con la consulta

- Tenemos una lista completa

```
http://es.php.net/manual/es/class.mysqli-result.php
```

Resumen

```php
$conexion= new mysqli($host,$user,$pass,$bd);
if ($conexion->connect_errno==null){
   $resultado = $conexion->query($consulta);
   $numFilas = $resultado->num_rows;
   $numCampos = $resultado->fields_count;
   echo "La consulta ha retornado $numFilas filas con  $numCampos columnas";
   $fila = $resultado->fetch_array();
   while ( $fila){
          echo"El valor del primer campo es $fila[0]";
          $fila = $resultado->fetch_array();
   }
   $resultado->free();
   $conexion->close();
}
```

Actividad

Haciendo una consulta del tipo select * from producto where pvp < 200, realiza un código que visualizce en una tabla los resultados

Transacciones

Definición

- Una transacción es un conjunto de acciones u operaciones que
- Esta se realizan contra una base de datos de modo que o bien se realizan correctamente todas
- o no se realiza ninguna

- Supongamos que hacemos una transferencia bancaria; cuando menos implica descontar dinera de una cuenta e ingresaro en otra

Transacciones

- Ahora supongamos que nada mas descontar el dinero de una cuenta se cae la luz y se apaga el servidor
- Esto creará una inconsistencia en la base de datos
- Por defecto en mysqli cada acción con la base de datos es una transacción en sí misma, pero esto se puede modificar

```
$conexion = new mysqli(..);
$conexion->autocommit(false);
.....
```

- Si se ha desactivado el autocommit, para terminar una transacción debemos usar los métodos ***commit*** o ***rollback***

```
$conexion = new mysqli(..);
$conexion->autocommit(false);
.....

if (CondicionOK){
   //Terminamos la transacción confirmando todas las acciones sobre la base de
   //datos desde que se inició la  transacción
   $conexion->commit();
}else{
   //Terminamos la transacción deshaciendo  todas
   //las acciones sobre la base de datos desde que se inició la
   //transaccion, y dejando la base de datos igual que estaba al principio
 $conexion->rollback();
 }
```

Actividad

Ejercicio de transacciones

Injecciones SQL

- Es un problema de seguridad importante, que hay que conocer y evitar
- Existe mucha documentación al respecto, en general podemos afirmar que un buen conocimiento de SQL proporcina herramientas tanto para poder establecer este tipo de ataques, como para podernos prevenir de ellos.
- Aportamos referenicas de web que nos pueden interesar consultar

Recursos de la Web

1. http://php.net/manual/es/security.database.sql-injection.php

2. http://es.wikipedia.org/wiki/Inyecci%C3%B3n_SQL
3. http://www.w3schools.com/sql/sql_injection.asp

Otras entradas un poco cuestionables por finalidad ???

Recursos de la Web

1. http://foro.elhacker.net/tutoriales_documentacion/tutorial_de_inyeccion_sql_sql_injection-t98448.0.html

1. http://www.mclibre.org/consultar/php/lecciones/php_db_inyeccion_sql.html#L2641
2. http://blog.netrunners.es/tabla-de-trucos-para-inyeccion-de-sql/

- A continuación y usando el ejemplo anterior de ***acceso*** vamos a probar a realizar un sencillo ataques sql.

1. Entrar en la plataforma sin tener acceso

Entrar en la plataforma sin tener acceso

- Entramos en una página y vemos el siguiente acceso

- Como no sabemos el usuario ni contraseña probamos a ver si se puede hacer una inserción no controlada
- Como programadores esperamos que en el código haya algo del estilo, como es nuestro caso

```
$nombre=$_POST['usuario'];
$pass=$_POST['pass'];

$consulta="select * from usuarios where nombre = \"$nombre\" and pass = \"$pass\" ";
$resultado = $conexion->query($consulta);
```

- Si todo fuera normal y nombre fuera por ejemplo "maría" la consulta que se envía al servidor sería

```
    select * from usuarios where nombre = "maria"
```

- Esta consulta si existe el usuario maría nos retornará una tupla, si no no devolverá ninguna.

Inyecciones sql

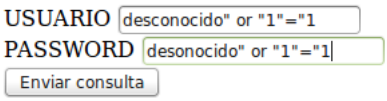
- Pero si añadimos más cosas obtendremos segura una respuesta, por ejemplo si en el codigo $nombre="maria or

\"1\"= \"1\" "

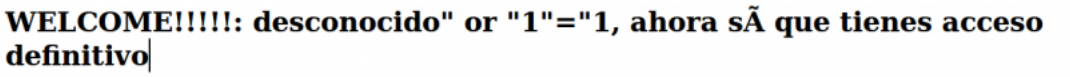
- Entonces la consulta quedaría

```
select * from usuarios where nombre = "maria" or "1"="1"
```

- Que nos devolverá todas las filas
- Así que si introducimos estos datos



- Entramos al sistema sin conocer usuario y contraseña



Consultas preparadas

- Una consulta preparada consiste en establecer una consulta como si fuera una variable y ejecutarla posteriormente tantas veces como sea necesario.
- Estas consultas se almacenan en el servidor y están listas para ser ejecutadas cuando sea necesario. El servidor solo tiene que analizarlas una vez
- Para trabajar con consultas preparadas, debemos usar la clase ***mysqli_stmt***, e inicializarla con el método ***stmt_init***

```
$conexion = new mysqli('localhost', 'dwes', 'abc123.', 'dwes');
//Preparo el objeto $consulta para crear consultas preparadas en él
$consulta = $conexion->stmt_init();
```

Los pasos para trabajar con consultas preparadas son:

1. Preparar la consulta en el servidor MySQL utilizando el método ***prepare***.
2. Ejecutar la consulta, tantas veces como sea necesario, con el método ***execute***.
3. Una vez que ya no se necesita más, se debe ejecutar el método ***close***.

```
$consulta = $conexion->stmt_init();
$consulta->prepare('INSERT INTO familia (cod, nombre) VALUES ("TABLET", "Tablet PC")');
$consulta->execute();
$consulta->close();
$conexion->close();
```

Parametrizar las consultas preparadas

- El uso real de las consultas preparadas es que los valores que pasas se asignen antes de ejectuar la consulta.
- La idea es preapara la consulta sin indicar los valores.

- Asignar los valores y ejectuar la consulta cuantas veces sea necesario.
- Veamos el proceso

Parametrizar la consulta

- Consiste en indicar en la consulta preparada en lugar de los valores, signos de interrogación ***?***
- En el caso anterior

```
$consulta->prepare('INSERT INTO familia (cod, nombre) VALUES (?,?);
```

- Ahora habría que asigar los valores. Para ello usamos el método ***bind_param'***

```
bind_param(tipoDatos, variables_con_los_valores)
```

- Este método recibe dos tipos de parámetros

1. El primero es una cadena de caracteres, donde cada carácter especifica el tipo de valor que va a recibir cada uno de los valores esperados en la consulta.

- - La codificación sería :

1. ***s***: cadena de caracteres
2. ***i***: número entero
3. ***d***: número float
4. ***b***: valor binario (BLOB)

Consultas preparadas

- En nuestro caso como va a recibir en los dos parámetros cada uno una cadena de caracteres sería ***"ss"***

1. El segundo grupo sería cada uno de los valores.SIEMPRE hay que especificar variables

- En el ejemplo que estamos siguiendo

```
$consulta = $conexion->stmt_init();
$consulta->prepare('INSERT INTO familia (cod, nombre) VALUES (?, ?)');
$cod_producto = "TABLET";
$nombre_producto = "Tablet PC";
$consulta->bind_param('ss', $cod_producto, $nombre_producto);
```

- Insisto en que siempre hay que especificar ***variables***, nunca directamente valores.
- Vemos el siguiente ejemplo:

```
$consulta->bind_param('ss', 'TABLET', 'Tablet PC');  // Genera un error
```

Resumen

```
$conexion = new mysqli(...);
$consulta = $conexion->stmt_init();
$consulta->prepare(...sentencia ... con ???)
$consulta->bind_param('s-i-b-d(tipo_de_valores)',
```

```
            valores_en_variables_respectivos_a_????');
$consulta->execute();
$consulta->close();
$conexion->close();
```

Consultas preparadas que retornan valores

- En caso de que la consulta preparada retorne valores se recogen con el método ***bind_result'***
- Este método recibirá variables en los que se almacenarán los valores
- Para recorre el conjunto de valores, usamos el método fectch()
- Vemos el siguiente ejemplo

Consultas preparadas que generan valores

```
$consulta = $conexion->stmt_init();
$consulta->prepare('SELECT producto, unidades FROM stock WHERE unidades<2');
$consulta->execute();
$consulta->bind_result($producto, $unidades);
while($consulta->fetch()) {
        print "<p>Producto $producto: $unidades unidades.</p>";
}
$consulta->close();
$conexion->close();
```

- Aquí hay un enlace para una información completa sobre consultas preparadas

```
http://php.net/manual/es/class.mysqli-stmt.php
```

Actividad

- Modifica el ejercicio anterior usando consultas parametrizadas

Práctica de tienda

- Vamos a trabajar con la base de datos de la tienda
- Lo primero usando la herramienta workbench generamos el modelo de tablas de la base de datos dwes y la analizamos
- Crea una página web en la que se muestre el stock existente de un determinado producto en cada una de las tiendas.

- Para seleccionar el producto concreto utiliza un cuadro de selección dentro de un formulario en esa misma página.

Práctica de tienda

- Puedes usar como base los siguientes ficheros css y plantilla adjuntos.
- Añade la opción de modificar el número de unidades del producto en cada una de las tiendas.
- Utiliza una consulta preparada para la actualización de registros en la tabla stock.
- No es necesario tener en cuenta las tareas de inserción
  - (no existían unidades anteriormente)
- Tampoco las de borrado (si el número final de unidades es cero).

Qué es PDO

- La extensión PDO (PHP Data Objects) permite acceder a diferentes gestores de bases de datos utilizando las misma funciones.
- Esto es una gran ventaja frente a la extensión vista anteriormente mysqli,
- PDO nos abstrae de forma completa el sistema gestor que se utiliza.
- Como comentamos en el tema anterior, necesitaremos el driver concreto dependiendo del sistema gestor de bases de datos.
- Esto es lo único que tendremos que cambiar en nuestro programa para que funcione en uno u otro gestor de bases de datos, sin tener que cambiar nada del sql.

- En PHP 5 existen drivers para acceder a las bases de datos más populares (MySQL, Oracle, MS SQL Server, PostgreSQL, SQLite, Firebird, DB2, Informix, etc).
- En el siguiente enlace podemos ver los controladores de PDO que soporta directamente php.

```
http://es.php.net/manual/es/pdo.drivers.php
```

- En esta lección se explica el acceso a MySQL y SQLite mediante PDO. La extensión PDO no evalúa la correción de las consultas SQL.

Establecer conexión con PDO

- Para establecer una conexión lo que hacemos es instanciar un objeto de la clase PDO

```
$conexion = new PDO(...);
```

El constructor tien 4 parámetros de los cuales sólo el primero es obligatorio

Origen de datos (DSN).

Este parámetro es un string que la información del controlador del driver que voy a utilizar y se especifica de la siguiente manera

```
controlador:parametro1=dato1;parametro2=datos...parametron=daton
```

Los parámetros a especificar dependerá del controlador que vamos a utilizar, en general me informarán del controlador del driver que voy a utilizar como por ejemplo el nombre o dirección IP del servidor, el nombre de la base de datos).
Por ejemplo en el caso del controlador mysql

```
$conexion = new PDO('mysql:host=localhost;dbname=dwes', ...);
```

Nombre de usuario
Contraseña del usuario.
Opciones de conexión, almacenadas en forma de array.

- Muchas de las opciones de conexión dependerán del driver que vayamos a utilizar
- Por ejemplo con mysql podemos verlas aquí http://php.net/manual/es/ref.pdo-mysql.php

```
(Ver dentro de cada página de controladores http://php.net/manual/es/pdo.drivers.php)
```

Conxión con mysql

- En el caso de mysql en parámetro DNS tendríamos los siguientes datos
- ***host*** Nombre o dirección IP del servidor.
- ***port*** Número de puerto TCP en el que escucha el servidor.
- ***dbname*** Nombre de la base de datos.
- ***unix_socket*** Socket de MySQL en sistemas Unix.
- Como podemos ver en el ejemplo anterior, no todos los datos del parámetro DNS son obligatorios, podemos establecer la conexión con ***host*** y ***dbname***.
- Respecto a las opciones de conexión permiten establecer varios cuestiones
- Una vez establecida la conexión se pueden consultar/acutalizar valores de opciones de la conexión usando los métodos

```
getAtribute(int $atributo);
setAtribute(int $atributo, mixed $valor);
```

- Podemos ver los atributos en la página
  http://es.php.net/manual/es/pdo.setattribute.php
  http://es.php.net/manual/es/pdo.getattribute.php

Realizar consultas con PDO

- En el caso de PDO, se diferencias las consultas que retornan datos (SELECT) y las que actúan sobre el contendio de los datos (INSERT, UPDATE, DELETE)

INSERT, OPDATE, DELETE

- En este caso la sentencia se ejecuta envíandola con el método ***exec($sentencia)***
- Este método retorna un entero que indica en número de registros afectados

```
$conexion= NEW PDO("mysql:host=localhost;db=dwes","root","root");
$registros = $conexion->EXEC("DELETE FROM stock WHERE unidades=0");
print "<p>Se han borrado $registros registros.</p>";
```

SELECT

- En este caso debemos usar el método de la clase PDO llamado ***query($consulta)***
- Este método retorna un objeto de la clase ***PDOStatement***
  http://es1.php.net/manual/es/class.pdostatement.php
- Una vez que tenemos el objeto de la clase ya tenemos ese cursor o conjunto de filas con su puntero
- Para extraer cada fila usamos el método ***fetch()***, el cual en caso de que no haya filas que retornar devuelve null (El mismo concepto trabajado hasta ahora).

- Cada vez que hacemos un fetch obtenemos un array con la fila que podemos usar tanto de forma asociativa como indexada.
- Este comportamiento por defecto se puede cambiar, es decir que podemos obligar a que el array que devuelve sea indexado, asociativo o que sea un objeto.
- Para ello debemos pasar al método fetch un valor que lo especifique según la lista siguiente.
- Para cerrar el cursor se emplea el método ***closeCursor()***; muchos gestores de bases de datos necesitas que se libere, antes de ser usado para realizar otra consulta.

1. PDO::FETCH_ASSOC. Devuelve solo un array asociativo.
2. PDO::FETCH_NUM. Devuelve solo un array con claves numéricas.
3. PDO::FETCH_BOTH. Devuelve un array con claves numéricas y asociativas. Es el comportamiento por defecto.
4. PDO::FETCH_OBJ. Devuelve un objeto cuyas propiedades se corresponden con los campos del registro.

- A continuación diferentes formas de hacer exactamente lo mismo

```php
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$resultado = $conexion->query("SELECT producto, unidades FROM stock");
while ($registro = $resultado->fetch()) {
    echo "Producto ".$registro['producto'].": ".$registro['unidades']."<br />";
}
```

```php
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$resultado = $conexion->query("SELECT producto, unidades FROM stock");
while ($registro = $resultado->fetch(PDO::FETCH_ASSOC)) {
    echo "Producto ".$registro['producto'].": ".$registro['unidades']."<br />";
}
```

```php
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$resultado = $conexion->query("SELECT producto, unidades FROM stock");
while ($registro = $resultado->fetch(PDO::FETCH_NUM)) {
    echo "Producto ".$registro[0].": ".$registro[1]."<br />";
}
```

```php
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$resultado = $conexion->query("SELECT producto, unidades FROM stock");
while ($registro = $resultado->fetch(PDO::FETCH_OBJ)) {
    echo "Producto ".$registro->producto.": ".$registro->unidades."<br />";
}
```

Resumen

```php
 $conexion="mysql:host=localhost;dbname=dwes";
$user="root";
$pass="root";
$opciones=array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8");

$conexion=new PDO($conexion,$usuario,$pass, $opciones);
$consulta = "Select * from ...";
$sentencia = "Insert into ....."
$resultado = $conexion->exec($sentencia);
$sentencia->closeCursor();
echo "Se han insertado $resultado filas";
$resultado = $conexion->query($sconsulta);
while $resultado->fetch(){
  echo "se la leído el valor $resultado[0], ...";
}
$conexion=null; //Es la manera de liberar a la memoria de este objeto.
```

Consultas preparadas

- Al igual que en mysqli, podemos preparar las consultas. Esta forma de trabajar es cómoda y mas segura que la habitual, según viemos en apartados anteriores
- Para realizar una consulta parametrizada, hay que seguir unos pasos al igual que en ***mysqli***

preparar la consulta *prepare(...)*

- Para ello se pueden pasar con ? los valores de los parámetros o bien poner un nombre precedido de :

```
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$consulta = $conexion->prepare('INSERT INTO familia (cod, nombre) VALUES (?, ?)');
```

Es igual que hacer

```
$conexion = new PDO("mysql:host=localhost;dbname=dwes", "dwes", "abc123.");
$consulta = $conexion->prepare('INSERT INTO familia (cod, nombre) VALUES (:codigoProducto, :nombreProducto
```

Asignar valores a la consulta pararmetrizada

- Si se han especificado ? se asigna dando a cada parámetro un valor con el método bindParam(posicion, valor)

```
$cod_producto = "TABLET";
$nombre_producto = "Tablet PC";
$consulta->bindParam(1, $cod_producto);
$consulta->bindParam(2, $nombre_producto);
```

- Si se han especificado con nombre se usan los nombre en lugar de los números

```
$cod_producto = "TABLET";
$nombre_producto = "Tablet PC";
$consulta->bindParam(":cod", $cod_producto);
$consulta->bindParam(":nombre", $nombre_producto);
```

- Se ejecuta con el método execute()
- Este método permite alternativamente suplir las asignaciones anteriores realizadas con el método bindParam, pasándole en un argumento meditante una array dicha asignación.
- El array utilizado será asociativo o con claves numéricas dependiendo de la forma en que hayas indicado los parámetros.
- En el primer caso

```
$parametros = array[":nombre","TABLET"];
$consulta->execute($parametros);
```

- En el segundo caso

```
$parametros = array(":cod" => "TABLET", ":nombre" => "Tablet PC");
$consulta->execute($parametros);
```

Actividad

- Realiza un pequeño programa en php que usando la extensión ***PDO***, realice las siguientes acciones

1. Se conecte a la base de datos ***dwes***
2. insertamos un nuevo eleemntos o tupla en la tabla producto
3. consultamos todos los productos y los visualizamos
4. Hacemos otra consulta parametrizada de todos los productos de la tabla stock de una determinada tienda
    1. Esta última acción primero usando ***bindParam*** y luego sin usarlo (pasando directamente el parámetro al método execute

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php
        $dns = "mysql:host=localhost; dbname=dwes";
        $user='root';
        $pass='root';
        $opciones= array( PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8");
        //Realizamos una conexión básico pdo
        $conexion = new PDO($dns, $user, $pass, $opciones);
        if ($conexion)
            echo "conexión realizada satisfactoriamente";
        else
            echo "ohhhh!!!! no se ha conectado";

        //Ahora planteamos una sentencia de tipo insert
        $sentencia = "insert into producto
 values('NEW_PRODUCTO12','NOMBRE_PRODUCTO','NOMBRE_CORTO','DESCRIPCION
 DESCRIPCION',10000,'MP3')";

        //Y planteamos también una sentencia select
        $consulta ="select nombre_corto from producto";
        $filas= $conexion->exec($sentencia);
        echo"Se ha insertado correctamene $filas";
        //$filas será un objeto del tipo PDOStatement
        $filas= $conexion->query($consulta);

        while ($fila=$filas->fetch()){
            echo "Se ha recuperado $fila[0]<br/>";
        }
        $filas->closeCursor();
        $conexon=null;
        //Ahora hacemos la consulta parametrizadas
            //usando un objeto de la clase PDOStatement
        //Hacemos el prepare
        $sentencia= "Select producto from stock where tienda  = :nom";

        $consulta = $conexion->prepare($sentencia);
        $tienda =3;

        //$consulta->bindParam(':nom',$tienda,PDO::PARAM_INT);
        //$consulta->execute();
        //Podemos usar la opción de antes o esta otra
        $consulta->execute(array(":nom"=>$tienda));
        //Ahora mostramos los resultados
        while ($fila=$consulta->fetch()){
            echo "Visualizo el producto $fila[0]<br/>";

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

Control de excepciones

- A partir de la versión 5 se introdujo en PHP un modelo de excepciones similar al existente en otros lenguajes de

programación:

1. El código susceptible de producir algún error se introduce en un bloque ***try - catch***.

```
try{

//Instrucciones que pueden lanzar una excepción y
//puedo capturar en tiempo de ejecución

}cathc(Exception $e){
   echo "Se ha producido una excepcion". $e->getMessage();
}
```

1. Cuando se produce algún error, se lanza una excepción utilizando la instrucción ***throw***.
2. Después del bloque try debe haber como mínimo un bloque catch encargado de procesar el error.
3. Si una vez acabado el bloque try no se ha lanzado ninguna excepción, se continúa con la ejecución en la línea siguiente al bloque o bloques catch.

Actividad

Haz un programa que si dividimos por cero pase una exepcion

```
try {
    if ($divisor == 0)
        throw new Exception("División por cero.");
    $resultado = $dividendo / $divisor;
}
catch (Exception $e) {
    echo "Se ha producido el siguiente error: ".$e->getMessage();
}
```

- PHP ofrece una clase base Exception para utilizar como manejador de excepciones.
- Esta clase implementa dos métodos generales que nos muestran información sobre la excepticon que se ha podido producir
- ***getMessage***. Devuelve el mensaje, en caso de que se haya puesto alguno.
- ***getCode***. Devuelve el código de error si existe.
- El caso de PDO define su propia clase de excepciones que deriva o hereada de la clase Exception
- Para el caso concreto de PDO, hay que configurar para que lance las excepciones, puediento esta configuración tomar los siguientes valores:

1. PDO::ERRMODE_SILENT. No se hace nada cuando ocurre un error. Es el comportamiento por defecto.

2. PDO::ERRMODE_WARNING. Genera un error de tipo E_WARNING cuando se produce un error.
3. PDO::ERRMODE_EXCEPTION. Cuando se produce un error lanza una excepción utilizando el manejador propio PDOException.

- Vamos a ver como se utiliza:
- Primero activamos las excepciones, y luego ya se pueden utilizar

```
$dwes = new PDO("mysql:host=localhost; dbname=dwes", "dwes", "abc123.");
$dwes->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
try {
    $sql = "SELECT * FROM stox";
    $result = $dwes->query($sql);
    //…
}
catch (PDOException $p) {
    echo "Error ".$p->getMessage()."<br />";
}
```

- En este caso que no existe la tabla nos diría

```
Error SQLSTATE[42S02]: Base table or view not found: 1146 Table 'dwes.stox' doesn't exist
```

- En el caso de mysqli usaríamos la clase mysqli_sql_exception que gestiona el tema de las excepciones

```
http://es.php.net/manual/es/class.mysqli-sql-exception.php
```

Actividad

- Realicemos un fichero de conexión a base de datos que contenga las siguientes funciones

1. ***conectar***($bd, $usuario,$password) Retornará un objeto de la clase PDO si todo ok
2. ***consulta($sentencia,$parametros)*** Retorna un objeto de la clase PDOStatement si todo ok,
    1. Recibe dos argumentos, un string que será la consulta parametrizada, y un array con los valores para cada parámetro
3. ***Insertar($tabla,$valores)***
    1. Recibe un string que es el nombre de tabla y un vector que serán los diferentes valores para cada campo de la tabla
    2. Suponemos que se pasan los valores ok, si no, capturamos la excepción

Usuario:ManuelRomero/proyecto/proyectoIternova/nagios

De WikiEducator
< Usuario:ManuelRomero | proyecto

ACCIONES DEL PROYECTO

Acciones del proyecto

Proyecto de colaboración con Iternova

Volver

Acceso

Para acceder a alos serviores que tengo montados

```
172.17.0.3/nagios3 nagiosadmin / nagiosadmin
172.17.0.4/nagios3 nagiosadmin / nagiosadmin
```

Nagios server

- Cada nagios tiene la app de php
- Está hubicada en /var/www/html/nagios-api
- En ella es importante setting/modules.ini
- Ahí es donde se pone el password para acceder por php al api, en nuestro caso admin/admin
- Estas son las credenciales que hay que aportar al dar de alta un nuevo servidor nagios

}}

Instalación

- Înstalo un nuevo docker con nagios

ficheros de configuración
como arrancar nagios

```
root@b6656d05a87d:/# sudo service nagios3 start
 * Starting nagios3 monitoring daemon nagios3                          [ OK ]
root@b6656d05a87d:/# service nagios-nrpe-server start
 * Starting nagios-nrpe nagios-nrpe                                    [ OK ]
root@b6656d05a87d:/#


Hay que instalar  instalar php5_curl en  el servidor de smartroadº
```

como invocarlo y ver su funcionamiento
especificar el usuario y password
Retomando todo en noviembre/2016

Creo un docker iternovaWeb

```
docker exec internovaWeb
docker start -t -i internovaWeb /bin/bash
```

instalo siguiendo la siguiente web

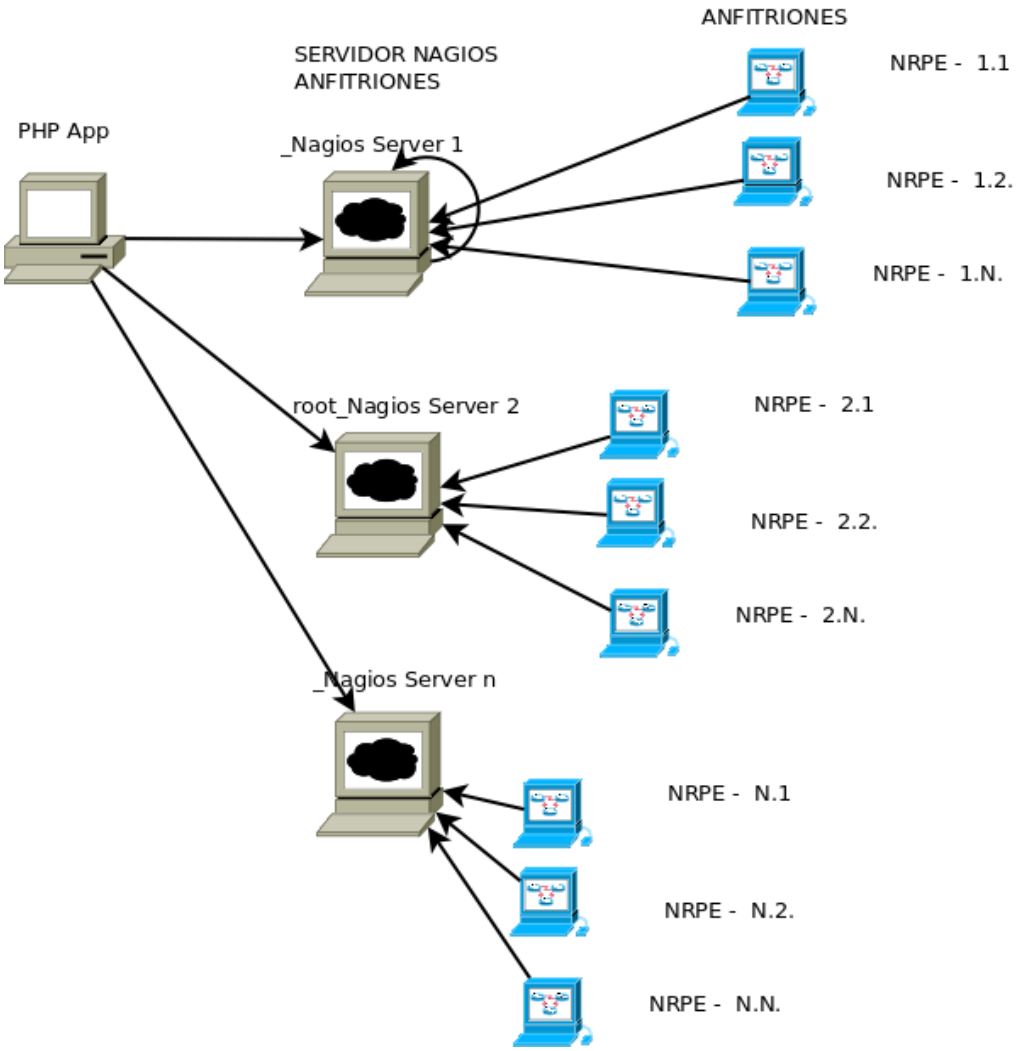
```
http://tecadmin.net/install-nrpe-on-ubuntu/#
sudo apt-get install nagios-nrpe-server nagios-plugins
```

- En la misma máquina instalo nagios

https://help.ubuntu.com/lts/serverguide/nagios.html

password ***nagios*** Selecciono que envíe por mail usando la máquina acutual como servidor de correo (no sé por qué hago esto, ya que no tiene sentido...)

- Esquema de nagios



- Tengo instalado nagios en la máquina 172.17.0.3, pero no consigo cargar la interfaz en el navegador ....

carga con 172.17.0.3/nagios3

nagios-api

- Tener en cuenta :
- instalar modulos rewrite y headers y cgi para su funcionamiento
- añadire allowOverride en el virtualhost del fichero de configuración de apache
  - Respecto a usuario y password
- Mirar el ./setting/modules.ini
- Usuario y pass de la sección api-server
- Actulizar ahí el url del servidor
- Tener en cuenta que no esté en la base de datos en la tabla parmas mapedados datos anterirores, en cuyo caso se borran los datos de la tabla para que vuelva a cargarlo
- Revisar la ruta cargada en el modules.ini de dónde toma los datos el nagios y actualizarlo en sección getternagios
- Y a trabajar con los datos

Nagios-api

Host a monitorizar

Servicios a monitorizar

Nombres que aparecen en el response del ...nagios-api/getternagios/data concretamente los service que es lo que me interesa

```
Estos son los nombres que tenemos descritos en /etc/nagios3/conf.d en los apartados service description
```

Servicio de uso de ssh SSH Servicio de uso web con http Servicio de uso de mysql Serivicio de uso de discos Current Load Servicio de uso de cpu CPU (Antes Total Process) Servicio de ping PING

Qué recoger de lo que me da el api

El api me da la siguiente información de cada servicio

CPU

```
["CPU"] =>        object(stdClass)#82 (54) {
        ["host_name"] =>          string(12) "7389fba856c3"
        ["service_description"] =>          string(3) "CPU"
        ["modified_attributes"] =>          string(1) "0"
        ["check_command"] =>          string(19) "check_procs!250!400"
        ["check_period"] =>          string(4) "24x7"
        ["notification_period"] =>          string(4) "24x7"
        ["check_interval"] =>          string(8) "5.000000"
        ["retry_interval"] =>          string(8) "1.000000"
        ["event_handler"] =>          string(0) ""
        ["has_been_checked"] =>          string(1) "1"
        ["should_be_scheduled"] =>          string(1) "1"
        ["check_execution_time"] =>          string(5) "0.007"
        ["check_latency"] =>          string(5) "0.231"
        ["check_type"] =>          string(1) "0"
        ["current_state"] =>          string(1) "0"
        ["last_hard_state"] =>          string(1) "0"
        ["last_event_id"] =>          string(1) "0"
        ["current_event_id"] =>          string(1) "0"
        ["current_problem_id"] =>          string(1) "0"
        ["last_problem_id"] =>          string(1) "0"
        ["current_attempt"] =>          string(1) "1"
        ["max_attempts"] =>          string(1) "4"
        ["state_type"] =>          string(1) "1"
        ["last_state_change"] =>          string(10) "1479883426"
        ["last_hard_state_change"] =>          string(10) "1479883426"
        ["last_time_ok"] =>          string(10) "1480327185"
```

```
        ["last_time_warning"] =>          string(1) "0"
        ["last_time_unknown"] =>          string(1) "0"
        ["last_time_critical"] =>          string(1) "0"
        ["plugin_output"] =>          string(22) "PROCS OK: 14 processes"
        ["long_plugin_output"] =>          string(0) ""
        ["performance_data"] =>          string(19) "procs=14;250;400;0;"
        ["last_check"] =>          string(10) "1480327185"
        ["next_check"] =>          string(10) "1480327485"
        ["check_options"] =>          string(1) "0"
        ["current_notification_number"] =>          string(1) "0"
        ["current_notification_id"] =>          string(1) "0"
        ["last_notification"] =>          string(1) "0"
        ["next_notification"] =>          string(1) "0"
        ["no_more_notifications"] =>          string(1) "0"
        ["notifications_enabled"] =>          string(1) "1"
        ["active_checks_enabled"] =>          string(1) "1"
        ["passive_checks_enabled"] =>          string(1) "1"
        ["event_handler_enabled"] =>          string(1) "1"
        ["problem_has_been_acknowledged"] =>          string(1) "0"
        ["acknowledgement_type"] =>          string(1) "0"
        ["flap_detection_enabled"] =>          string(1) "1"
        ["failure_prediction_enabled"] =>          string(1) "1"
        ["process_performance_data"] =>          string(1) "1"
        ["obsess_over_service"] =>          string(1) "1"
        ["last_update"] =>          string(10) "1480327366"
        ["is_flapping"] =>          string(1) "0"
        ["percent_state_change"] =>          string(4) "0.00"
        ["scheduled_downtime_depth"] =>          string(1) "0"
    }
```

- De ello de momento me voy a quedar con

```
["check_command"] =>          string(19) "check_procs!250!400"
["check_execution_time"] =>          string(5) "0.007"
["current_state"] =>          string(1) "0"
["plugin_output"] =>          string(22) "PROCS OK: 14 processes"
["long_plugin_output"] =>          string(0) ""
["performance_data"] =>          string(19) "procs=14;250;400;0;"
["last_check"] =>          string(10) "1480327185"
["next_check"] =>          string(10) "1480327485"
```

Obtenido de «http://es.wikieducator.org/index.php?title=Usuario:ManuelRomero/proyecto/proyectoIternova/nagios&oldid=22010»

Usuario:ManuelRomero/proyecto/proyectoIternova/Servicios

De WikiEducator
< Usuario:ManuelRomero | proyecto

ACCIONES DEL PROYECTO

Acciones del proyecto

Proyecto de colaboración con Iternova

Volver

1.- en ***Monitorizacion_Constants***

- Defino como constantes cada servicio a monitorizar
- En este caso tengo los siguientes

```
//Tipos de Servicios a monitorizar
   const SERVICIO_UNDEFINED = -1; //Servicio no definido
   const SERVICIO_SSH = 1; //SSH ssh.cfg
   const SERVICIO_HTTP = 2; //Servicio http HTTP
   const SERVICIO_MYSQL = 3; //Servicio uso mysql Mysql
   const SERVICIO_DISCOS = 4; // Current_load  Disk
   const SERVICIO_CPU = 5; //Servicio uso de cpu Load
   const SERVICIO_PING = 6;  //PING
   const SERVICIO_PROCESS = 7;  //Procesos  check_total_process
```

2.- En el servidor Nagios he de tener configurado los equipos que voy a monitorizar así como los servicios que quiero. En la siguiente carpeta

- En ***/etc/nagios3/conf.d#*** tengo los siguientes ficheros

contacts_nagios2.cfg generic-service_nagios2.cfg services_nagios2.cfg extinfo_nagios2.cfg host_monitorizar.cfg servicios.cfg generic-host_nagios2.cfg hostgroups_nagios2.cfg timeperiods_nagios2.cfg

- Importante observar el fichero host_monitorizar.cfg (host que quiero monitorizar) y servicios.cfg (Los serivicios a monitorizar).

3.- Los servicios que tenemos disponibles para monitorizar los disponemos en el servidor nagios

```
/usr/lib/nagios/plugins# ls
check_apt        check_icmp           check_nntps      check_smtp
check_breeze     check_ide_smart      check_nrpe       check_snmp
check_by_ssh     check_ifoperstatus   check_nt         check_spop
check_clamd      check_ifstatus       check_ntp        check_ssh
check_cluster    check_imap           check_ntp_peer   check_ssmtp
check_dbi        check_ircd           check_ntp_time   check_swap
check_dhcp       check_jabber         check_nwstat     check_tcp
check_dig        check_ldap           check_oracle     check_time
check_disk       check_ldaps          check_overcr     check_udp
check_disk_smb   check_load           check_pgsql      check_ups
check_dns        check_log            check_ping       check_users
check_dummy      check_mailq          check_pop        check_wave
check_file_age   check_mrtg           check_procs      negate
check_flexlm     check_mrtgtraf       check_real       urlize
check_ftp        check_mysql          check_rpc        utils.pm
check_host       check_mysql_query    check_rta_multi  utils.sh
check_hpjd       check_nagios         check_sensors
check_http       check_nntp           check_simap
```

- Aquí seleccionaríamos los servicios que queremos monitorizar
- Por otro lado, aquí una url dónde poder monitorizar todo

https://www.monitoring-plugins.org/doc/man/

2.- En el equipo que esté monitorizando ese servicio comprobar el script o configuración correspondiente en ***/etc/nagios-plugin/config*** tendrá que haber un cfg por servicio

```
ls
apt.cfg   disk.cfg   ftp.cfg   load.cfg  news.cfg  ping.cfg   real.cfg  tcp_udp.cfg  users.cfg
dhcp.cfg  dummy.cfg  http.cfg  mail.cfg  ntp.cfg   procs.cfg  ssh.cfg   telnet.cfg
```

Ficheros de configuración del nagios

```
/etc/nagios3/connf.d/servicios.cfg
```

```
define service{
        use                     generic-service  ; Name of service template to use
        host_name               5c43117b78e8 ;hay que poner cada host a monitorizar
        service_description     SSH
        check_command           check_ssh
        notifications_enabled 0
        }

define service{
        use                             generic-service         ; Name of service template to use
        host_name                       5c43117b78e8
        service_description             HTTP
        check_command                   check_http
        }

define service{
        use                             generic-service         ; Name of service template to use

        host_name                       5c43117b78e8
        service_description             MYSQL
        check_command                   check_mysql
        notifications_enabled           0
        }
define service{
        use                             generic-service         ; Name of service template to use
        host_name                       5c43117b78e8
        service_description             Disk
        check_command                   check_disk


        }


define service{
        use                             generic-service         ; Name of service template to use
        host_name                       5c43117b78e8
        service_description             Load
                check_command                   check_load!5.0!4.0!3.0!10.0!6.0!4.0
        }

define service{
        use                 generic-service   ; Name of service template to use
        host_name                       5c43117b78e8
        service_description             PING
        check_command                   check_ping!100.0,20%!500.0,60%
        }
define service{
        use                             generic-service         ; Name of service template to use

        host_name                       5c43117b78e8
        service_description             Process
        check_command                   check_procs
        notifications_enabled           0
        }
```

Usuario:ManuelRomero/proyecto/proyectoIternova/Crontab

De WikiEducator
< Usuario:ManuelRomero | proyecto

ACCIONES DEL PROYECTO

Acciones del proyecto

Proyecto de colaboración con Iternova

Volver

Acciones para activar crondeamon

1. En el shell mirar las importates, est un history de lo hecho

```
crontab -e -u www-data
 778  php  /var/www/smartroads-core/core/crondaemon/crontab.php
 779  gearmand -d
 780  php  /var/www/smartroads-core/core/crondaemon/crontab.php
 781  cd ../../logs/
 782  l
 783  tail -f crondaemon.log
 784  ls
 785  tail -f crondaemon.log
 786  killall -9 php; killall -9 gearmand
 787  gearmand -d
 788  tail -f crondaemon.log
 789  servivce cron start
 790  servivce crond start
 791  service crond start
 792  service cron start
 793  service cron status
 794  status cron
 795  tail -f crondaemon.log
```

1. En /core/crondeamon/setting/cronjog.ini añadimos

```
[monitorizacion]
file="modules/monitorizacion/monitorizacion_controller.php"
method="Monitorizacion_Controller::crondaemon";
```

1. cargar el demonio gearman (ver history)

```
 gearmand -d
```

- Tengo un problema con cron, no se ejecuta de forma periódica
- Mirando en internet hago lo siguiente

```
apt-get purge upstart
apt-get install upstart
apt-get install cron
```

- Daba un error en el proceso de instalación de cron, así que mirando vi esto

```
https://askubuntu.com/questions/365911/why-the-services-do-not-start-at-installation
```

- Lo que hice fue actulizar el fichero

```
vim  /usr/sbin# vim policy-rc.d
quitando el 101 del exits y poniendo 0
```

- Ahora veo que cron no se ejecuta como un servicio, simplemente escribiendo cron

```
cron
```

- Miro el log y no veo nada

tail -f /var/www/smartroads/logs/crondaemon.log </source>

Usuario:ManuelRomero/proyecto/proyectoIternova/Crontab

De WikiEducator
< Usuario:ManuelRomero | proyecto

ACCIONES DEL PROYECTO

Acciones del proyecto

Proyecto de colaboración con Iternova

Volver

Acciones para activar crondeamon

1. En el shell mirar las importates, est un history de lo hecho

```
crontab -e -u www-data
 778  php  /var/www/smartroads-core/core/crondaemon/crontab.php
 779  gearmand -d
 780  php  /var/www/smartroads-core/core/crondaemon/crontab.php
 781  cd ../../logs/
 782  l
 783  tail -f crondaemon.log
 784  ls
 785  tail -f crondaemon.log
 786  killall -9 php; killall -9 gearmand
 787  gearmand -d
 788  tail -f crondaemon.log
 789  servivce cron start
 790  servivce crond start
 791  service crond start
 792  service cron start
 793  service cron status
 794  status cron
 795  tail -f crondaemon.log
```

1. En /core/crondeamon/setting/cronjog.ini añadimos

```
[monitorizacion]
file="modules/monitorizacion/monitorizacion_controller.php"
method="Monitorizacion_Controller::crondaemon";
```

1. cargar el demonio gearman (ver history)

```
 gearmand -d
```

- Tengo un problema con cron, no se ejecuta de forma periódica
- Mirando en internet hago lo siguiente

```
apt-get purge upstart
apt-get install upstart
apt-get install cron
```

- Daba un error en el proceso de instalación de cron, así que mirando vi esto

```
https://askubuntu.com/questions/365911/why-the-services-do-not-start-at-installation
```

- Lo que hice fue actulizar el fichero

```
vim  /usr/sbin# vim policy-rc.d
quitando el 101 del exits y poniendo 0
```

- Ahora veo que cron no se ejecuta como un servicio, simplemente escribiendo cron

```
cron
```

- Miro el log y no veo nada

tail -f /var/www/smartroads/logs/crondaemon.log </source>

Usuario:ManuelRomero/proyecto/proyectoIternova/phpUnit

De WikiEducator
< Usuario:ManuelRomero | proyecto

ACCIONES DEL PROYECTO

Acciones del proyecto

Proyecto de colaboración con Iternova

Volver

Instalación

https://phpunit.de/getting-started.html

```
wget https://phar.phpunit.de/phpunit.phar
chmod +x phpunit.phar
sudo mv phpunit.phar /usr/local/bin/phpunit
phpunit --version
   PHPUnit 6.1.0 by Sebastian Bergmann and contributors.
```

Usuario:ManuelRomero/iternova

De WikiEducator
< Usuario:ManuelRomero

Contenido

Proyecto en iternova

Páginas para estar conectado

Slack

https://iternova.slack.com (Con lo de siempre)

git

https://gitlab.iternova.net/smartroads/smartroads-core (correo...)

Instalación del sistema

- Web de referencia

```
https://wiki.iternova.net/doku.php?id=development:servers:howto_server_sgwc
```

- Instalo vagrant
  - Para ver versión de sistema operativo instalado

```
uname -a  o lsb_release -a
```

- Preparao un sistema por defecto con puphpet.net (Apache, mysql. mongo, php6) y caso todo lo demás por defecto
- Descargo el fichero puphpet.zip y lo descomprimo
- ***vagrant up*** y me instala el sistema (una media hora)
- ***vagrant ssh*** y conecto con la máquina
- Empiezo a instalar cosas

libgearman

1. ***gearman*** => Un demonio para ejecutar tareas pesadas de la aplicación web en background

```
sudo apt-get install gearman gearman-job-server gearman-tools libgearman-dev
referencia https://wiki.iternova.net/doku.php?id=smartportal:modules:core:gearman (ManuelRomero x.....)
```

php-apc

- ***APC***: *Advanced PHP Cache*
- Instalacion

```
apt-get install php-apc
```

- Si aparece una línea tal que

```
PHP Warning:  apc_sma_info(): No APC SMA info available.  Perhaps APC is disabled via apc.enabled?,
```

- entonces en */etc/php5/mods-enabled/apcu.ini* añadimos las siguientes líneas, para garantizar que está activo APC tanto para Apache (*mod_php*) como para CLI (*cli*):

```
apc.enable_cli=1
apc.enable=1
extension=apcu.so
```

memcache

https://wiki.iternova.net/doku.php?id=development:servers:howto_memcache

```
sudo apt-get install memcached
sudo pecl install memcache
```

gd

Bibglioteca para crear y manipular imágenes y transferirlas desde el cliente al servidor

```
sudo apt-get install php5-gd
```

Referencia

```
http://php.net/manual/es/book.image.php
```

curl

```
http://php.net/manual/es/book.curl.php
```

- Para instalarlo

```
sudo apt-get install php5-curl
```

bzip

- Viene instalado por defecto

```
sudo apt-get install bzip2
```

- Para verificar la versión (iternova pide versión 1.0.5 o superior)

```
bzip --version o bzip -V
```

- Versión instalada 1.0.6 => OK!

json

- Ya instalada por defecto

```
sudo apt-get install php5-json
```

mbstring

- No consigo encontrar el paquete
- Veo que forma parte del buit-in de libapache2-mod-php. instalo este paquete

```
sudo apt-get install libapache2-mod-php5
```

- Y todo correcto

```
http://serverfault.com/questions/455388/how-to-install-php-xml-and-php-mbstring-on-php-5-4-9-4
http://ppa.launchpad.net/ondrej/php5/ubuntu/dists/precise/main/binary-i386/Packages
```

mongo

- Para instalar mongo como voy a usar pear, y veo que lo necesitaré y también debug, voy a instalar estos dos paquetes

```
sudo apt-get install php5-dev php-pear
```

- Después sigo los instrucciones del slack de iternov

```
https://wiki.iternova.net/doku.php?id=smartportal:howto:howto_model_and_database_mongodb
```

- Ahora instalo mongodb

```
apt-get install mongodb;
```

- Instalación de extensión para PHP

```
pecl install mongo;
```

- Durante la instalación pregunta

```
Build with Cyrus SASL (MongoDB Enterprise Authentication) support? [no] :
```

- Dejo por defecto el no, no se si lo necesitaré. No anoto para tenerlo en cuenta
- Al terminar me indica que modifique el php.ini

```
You should add "extension=mongo.so" to php.ini
```

- Lo hago según me indica
- Referenica web

```
http://docs.mongodb.org/ecosystem/drivers/php/
```

simplexml

- Instalado por defecto a partir de php 5.2
- Lo compruebo con

```
sudo php -m | grep SimpleXML
```

zip

```
pecl install zip
```

- Tras instalar me da un error que no encuentra pcr....
- Busco en internet y veo que debo de instalar esta librería

```
sudo apt-get install libpcre3-dev
```

- Tras hacerlo vuelvo a lanzar la instalación del zip
- Todo va ok!
- Después de instalar aparece el siguiente texto

```
Build process completed successfully
Installing '/usr/lib/php5/20131226/zip.so'
install ok: channel://pecl.php.net/zip-1.12.5
configuration option "php_ini" is not set to php.ini location
You should add "extension=zip.so" to php.ini
```

- Y procedemos a hacerlo
- Después lo quito ya que observo que por defecto zip ya está instalado

HTMLpurifier

- Quita código malicios, realiza escape de caracteres no deseados, evita inyecciones ....
- Web de referencia de HTMLPurifier

```
http://htmlpurifier.org
```

- Instalo:

```
pear channel-discover htmlpurifier.org
pear install hp/HTMLPurifier
```

Instalar nagios

http://www.nagios.org/

- Voy a la instalación con ubuntu

```
http://nagios.sourceforge.net/docs/3_0/quickstart-ubuntu.html
```

- Empiezo creando un usuario nagios/nagios

```
[08:40 AM]-[vagrant@packer-virtualbox-iso-1422643551]-[~]
$ sudo useradd -m -s /bin/bash nagios

[08:40 AM]-[vagrant@packer-virtualbox-iso-1422643551]-[~]
$ passwd nagios
passwd: You may not view or modify password information for nagios.

[08:40 AM]-[vagrant@packer-virtualbox-iso-1422643551]-[~]
$ sudo passwd nagios
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

- Creo el grupo ***nagcmd*** y añado los usuarios **nagios** y **www-data**
- Descargamos nagios y los plugins

```
wget http://prdownloads.sourceforge.net/sourceforge/nagios/nagios-3.2.3.tar.gz
wget http://prdownloads.sourceforge.net/sourceforge/nagiosplug/nagios-plugins-1.4.11.tar.gz
```

- Al hacer esto veo que el plugin no lo baja, lo descargo de otra página

```
wget http://nagios-plugins.org/download/nagios-plugins-2.0.3.tar.gz
```

- Una vez descargados los fuentes procedemos a descomprimir, compilar e instalar
  - Descomprimir

```
tar xzf nagios-3.2.3.tar.gz
```

Compilar

```
  cd nagios-3.2.3/
  ./configure --with-command-group=nagcmd
  make all
```

Instalar

```
 make install-init it

 make install-init

 make install-config

 make install-commandmode
```

- Ahora procedemos a configurar

```
vim /usr/local/nagios/etc/objects/contacts.cfg
```

- Básicamente te pide cambiar el usuario, el grupo y el correo para notificaciones
- Yo de momento he modificado lo siguiente

```
use nagios
email manuelromeromiguel@gmail.com
.....
members nagcmd
```

- instalamos y configuramos el interfaz web
- Instalar un fichero nagios.conf en el /etc/apache2/conf.d

```
make install-webconf
```

- Ponemos una password para el usuario nagiosadmin

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Ahora instalamos el pluging de nagios

- Extreemos el contenido del fichero descargado, lo compilamos e instalamos

```
tar xzf nagios-plugins-1.4.11.tar.gz
cd nagios-plugins-1.4.11
./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
make install
```

Iniciar nagios

- Una vez instalado procedemos a ponerlo en marcha

Problemas varios con nagios

- Con consigo arrancarlos y analizando problemas

Miro los diferentes directorios

1. conf => nagios.con /etc/apache2/conf.d/nagios.conf
2. web => /usr/local/nagios/share/index.php
3. usuario => /usr/local/nagios/etc/htpasswd.user
4. cgi => /usr/local/nagios/sbin

- He instalado fcgiwrap pero creo que eso es para ejecutar cgi con nginx, que no tiene que ver con apache
- He cambiado los el propietario del fichero htpasswd.user que estaba a root y se lo he puesto a nagios, pero creo que no tiene nada que ver ...
- HE mirado en internet y no he encontrado nada claro

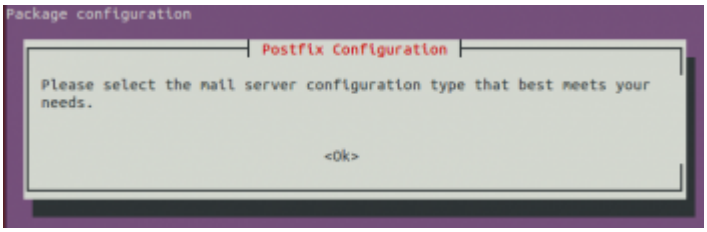
Alternativas

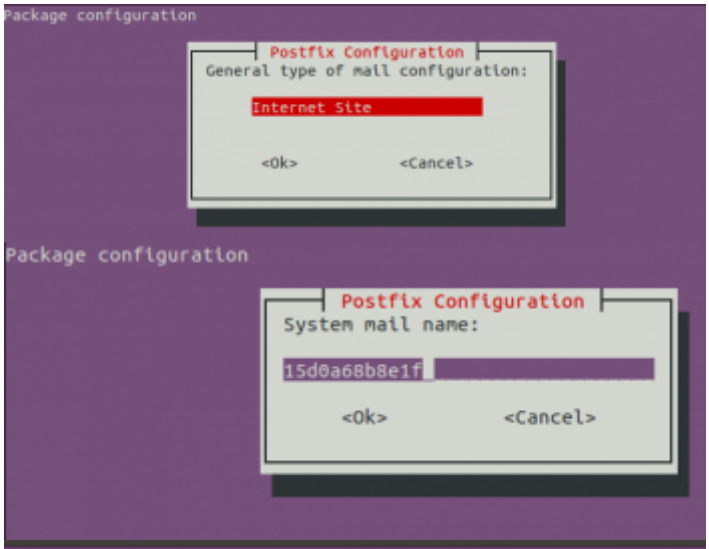
https://www.howtoforge.com/installing-nagios-on-debian-lenny-and-monitoring-a-debian-lenny-server

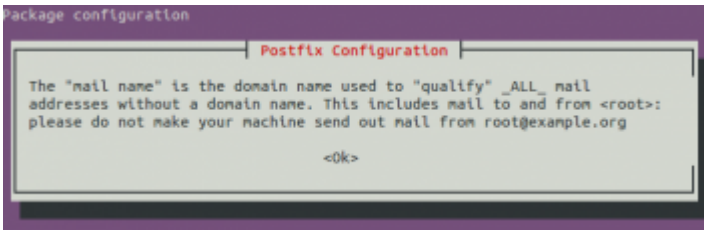
- Instalo nagios desde el repositorio

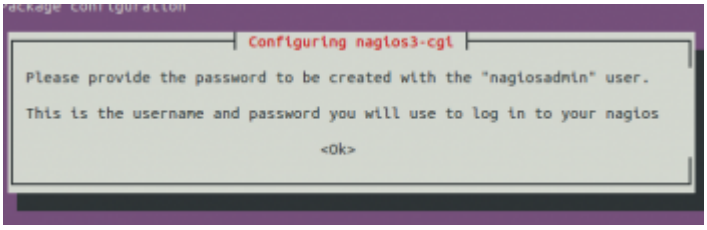
sudo apt-get install nagios3 nagios-plugins nagios-nrpe-plugin

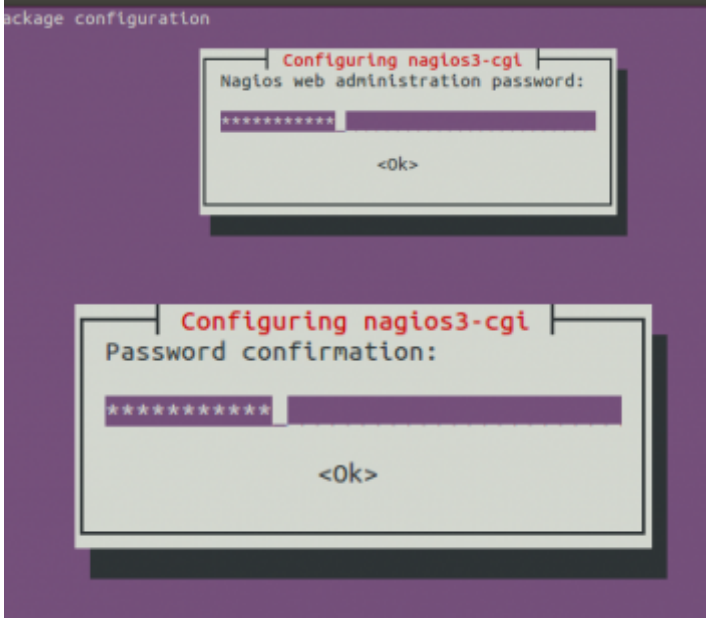
- En la instalación obtengo los siguientes pantallazos

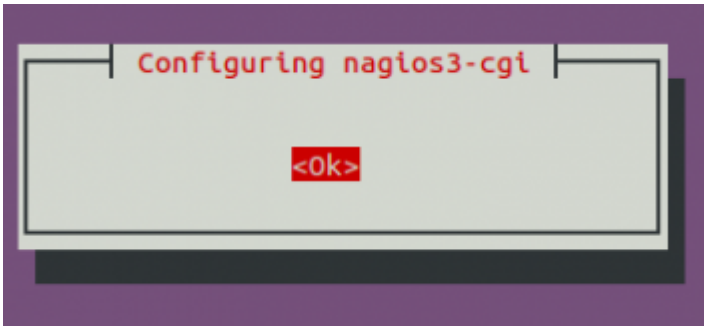












Centreon

Sigo la siguiente web para la instalacion

```
https://operativoslinux.wordpress.com/2015/01/03/nagios-centreon-guia-de-instalacion-y-configuracion/
```

- Primero instalamos las dependencias

```
aptitude install apache2 php5 php5-mysql php-pear php5-ldap php5-snmp php5-gd mysql-server-5.5 libmysqlclient-d
```

- Me da un error que no puede encontrar la librería ***libgd2-xpm***
- Como no veo solución directamente la quito y se instala todo ok. Dejo este tema pendiente

Durante la instalación aparecen los siguientes pantallazos Archivo:CenteronInstalacion1.png Archivo:CenteronInstalacion2.png Archivo:CenteronInstalacion3.png Archivo:CenteronInstalacion4.png Archivo:CenteronInstalacion5.png Archivo:CenteronInstalacion6.png Archivo:CenteronInstalacion7.png Archivo:CenteronInstalacion8.png Archivo:CenteronInstalacion9.png

- Al final me da un error y aborta
- Arranco el gestor de base de datos y vuelvo a ejecutar todos los paquetes

```
sudo service mysql restart (estaba apagado)
```

- Y de nuevo hago la instalación anterior, y todo ok (lógicamente no me vuelve a instalar todos los paquetes
- Después de la instalación recibo continuamente un ***forbidden***
- Mirando en internet realizo un cambio que no veo muy bien, pero funciona

Ên el fichero centreon.conf (/etc/apache2/conf-enable/centreon.conf

- donde había

```
Order allow,deny
Allow from all
```

- Lo comento y pongo otra línea

1. Order allow,deny
2. Allow from all

```
 Require all granted
```

- Así funciona

..Entro en la configuración Se me queda colgado intentao acceder a la base de datos intengo varias consas my.conf comento el bind_address

En hosts comento eh 172.... con "dafdasdfasdf"

En mysql ejecuto la siguiente sentencia

- Al final creo un nuevo usuario que no sea root y le doy privilegios totales (idea de jorge)

```
GRANT ALL PRIVILEGES  ON *.* TO manolo@'%' identified by 'manolo'
```

- Y todo ok.

- Creo una imangen con el contenedor actual

```
docker commit -m "iternova Sistema" -a "manolo" d6e769a6fac0 manolo/ubuntu-apache-nagios-centreon
```

- El sistema me dice

```
c99e0086d5805cfd7d24090e54c3603b8b794a54b33a76a302e86a987af13911
```

Estado

```
docker commit -m "iternova Apache2 Centreon" -a "manolo" d6e769a6fac0  manolo/ubuntu-apache
```

Preguntas del sistema

1.-

```
Do you accept GPL license ?
[y/n], default to [n]:
> y
```

2.-Do you want to install : Centreon Web Front

[y/n], default to [n]: > y

3.-Do you want to install : Centreon CentCore

[y/n], default to [n]: > y

4.-Do you want to install : Centreon Nagios Plugins

[y/n], default to [n]: > y

5.-Do you want to install : Centreon Snmp Traps process

[y/n], default to [n]: > y

6.-Where is your Centreon directory?

default to [/usr/local/centreon] >

7.-Do you want me to create this directory ? [/usr/local/centreon]

[y/n], default to [n]: > y

8.-

Where is your Centreon log directory default to [/usr/local/centreon/log] > y

9.-Do you want me to create this directory ? [/usr/local/centreon/log]

[y/n], default to [n]: > y

10.-Where is your Centreon etc directory

default to [/etc/centreon] >

11.-Do you want me to create this directory ? [/etc/centreon]

[y/n], default to [n]: > y

12.-Where is your Centreon binaries directory

default to [/usr/local/centreon/bin] >

13.-Do you want me to create this directory ? [/usr/local/centreon/bin]

[y/n], default to [n]: > y

14.-Where is your Centreon data informations directory

default to [/usr/local/centreon/data] >

15.-Do you want me to create this directory ? [/usr/local/centreon/data]

[y/n], default to [n]: > y

16.-Where is your Centreon variable library directory?

default to [/var/lib/centreon] >

Do you want me to create this directory ? [/var/lib/centreon] [y/n], default to [n]: > y

17.-Where is PEAR [PEAR.php]

default to [/usr/share/php/PEAR.php] >

18.-Path /usr/share/php OK

/usr/bin/perl OK Enable Apache configuration OK ERROR: Conf centreon does not exist!
Finding Apache user : www-data Finding Apache group : www-data

What is the Centreon group ? [centreon] default to [centreon] > y

Do you want me to create this group ? [centreon] [y/n], default to [n]: > y

What is the Centreon user ? [centreon] default to [centreon] >

Do you want me to create this user ? [centreon] [y/n], default to [n]: >

What is the Monitoring engine log directory ? > log

Where is your monitoring plugins (libexec) directory ? default to [/usr/lib/nagios/plugins] >

- Tengo que actualizar todo esto
- para dar permisos a root en remoto

```
GRANT ALL PRIVILEGES on *.* TO 'root'@'%' IDENTIFIED BY 'root' WITH GRANT OPTION
FLUSH  PRIVILEGES;
```

- Y parece que todo chuta
- Al final no consigo entrar en centreon, continuamente está como queriendo volver a instalar
- Renombre el directorio install y todo ok!.

Otros tema

- Centreon
- qué es nagios
- snmp para copiar datos de dispositivos
- Instalar dispositivos con nagios para controlar ?????
- Para ello sigo la página https://help.ubuntu.com/community/Nagios3
- Me da un fallo y vuelvo a instalar de esta otra página

Trabajando con la nueva configuración de vagrant

Se usa fpm para php que es un servicio independiente de apache... apt-get service php5-fpm restart

- ME aparece un problema para conectarme a nagiso

LEo que tengo que dar un valor a request_terminate_timeout /etc/php5/fpm/pool.d/www.conf

- En ese fichero modifico y escribo
- Ponía:

```
;request_terminate_timeout=0
```

- Pongo

```
request_terminate_timeout= ;request_terminate_timeout=0
```

Poniendo datos

git veo que me falta de instalar git

```
apt-get install git
```

nagios nagiosadmin / nagiosAdmin centreon admin/admin (bd manolo/manolo) mysql root/root manolo/manolo

bajo el proyecto

```
git clone https://gitlab.iternova.net/smartroads/smartroads-core.git
```

El proyecto se instala en $HOME/smartroadas-core

- Creo un enlace directo en mi DocumentRoot

```
ln -s /home/manolo/smartroads-core /var/www/html/smartroad
```

- Ahora intento cargar la página y me no me sale nada
- En el log de apache me sale el error relacionado con memcacne
- Veo que lo que me falta es instalar memcache en el lado del cliente, para que lo use php

```
pecl install memcache
```

Modificamos el php.ini, tal cual nos indican en la instalacion, añadiendo extensio=memcache.so

Ahora nos da un error de mongo

```
ERROR MongoDB: Error connecting to MongoDB server: Failed to connect to: localhost:27017: Connection refused
```

Recordamos

- Modificamos el fichero web.ini

```
HTTP_ADDRESS
HTTO_ADDRESS_CONTROLLER_SERVER
```

- El fichero /etc/mongodb.conf
  - descomentamos la línea

```
auth=true
```

añadimos el usuario en mongo db.addUser("root","root"); db.auth("root","root");

En el directorio creo las carpetas

```
files
cache
```

- Doy permisos de escritura a files, cache y logs
- Tras modificar abrir mongodb con rockmongodb y eliminar la colección del win.inif, para que cargue los datos

```
http://172.17.0.1/smartroad/admin/config/rockmongo/i
```

- Rebotar también el servicio de memcache.

Instalando las bases de datos de mongo y mysql

Creo el usuario en la base de datos de mongo use admin db.auth("root","root"); use smartroads_core db.auth.....

instalo la base de datos Para ello descargo el fichero en el local y lo descomprimo Posteriormente lo cargo en la base de datos con el comando mongorestore

```
mongorestore --username ....... --password ....  --db smartroads_core dump/smartroads/
```

mysql

- Descargo el ficheor sql de smartroads y lo cargo

```
mysql -u root -pxxxx < smartroadsxxxxxxxx.sql
```

Empezando el proyecto

- Creo un módulo nuevo llamado ***monitorizacion*** basado a partir del módulo smartapps

Copiamos todo cambiamos el nombre

smartapp => monitorizacion
SmartApp => Monitorizacion

29-4-2016

- Instalo rokcmongo una especie de phpmyadmin pero para db de mongo

```
Then, download Rockmongo from web
http://rockmongo.com/downloads
Extract to /var/www/ folder
unzip rockmongo-.zip /var/www/
```

- Configuramos la usuario admin en mongo para gestionar la base de datos desde mongo
- Modificamos el fichero ini de mongo, para que sea necesario autentificarse a la hora de acceder a una base de datos

```
vi /etc/mongodb.conf
Y añadir (o descomentar) la línea:

auth = true
```

- Creamos un usuario para la base de datos admin (usuario root)

```
$ mongo
 use admin
 db.addUser("root", "password")
```

- Ahora nos autentificamos para poder hacer cosas sobre la base de datos

```
db.auth("root","root")
```

Usuario:ManuelRomero/proyecto/proyectoIternova/DockerBase

De WikiEducator
< Usuario:ManuelRomero | proyecto

Contenido

Docker con nagios

- Descargo el DockerFile de https://github.com/cpuguy83/docker-nagios
- Creo la imagen con el tag nagios/iternova

```
sudo docker build -t nagios/iternova .
```

- Una vez creado el contenedor observo

```
sudo docker images
```

```
{10:59}~/proyecto/docker/nagios ➪ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nagios/iternova     latest              1d76d9cd011c        About a minute ago  536.2 MB
```

- Ahora arranco la imagen u creo un contenedor, puedo añadir un volumen

```
sudo docker run --name iternova -v /var/www/iternova:/var/www -dti iternova/web
```

Crear una imagen a partir de un contenedor

- Primero busco el id del contaniner del que quiero crear un contenedor

```
 {0:11}~ ➪ sudo docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                      PORTS               NAMES
7cf1f94fed51        nagios              "/bin/bash"              5 days ago          Up 37 hours                                     nagios
10efc1e40951        0e109c1a6cc7        "/bin/sh -c 'htpsswd "   5 days ago          Exited (127) 5 days ago                         sick_swirles
```

- En este caso la que nos interesa es ***7cf1f94fed51***
- Ahora creamos la imagen

```
 sudo docker commit -m="Instalando nagios" -a="Manuel Romero" 7cf1f94fed51 iternova/nagios:v0.1
[sudo] password for manuel:
sha256:479d0fed8381d22d953ad5ac18cc73a63dc8f812b4362365442b7a8807f26fcc
{0:06}~ ➪
```

- Ahora comprobamos las imágenes creadas

```
{0:06}~ ➪sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
iternova/nagios     v0.1                479d0fed8381        44 seconds ago      534.7 MB
....
```

Vemos que hemos creado la imagen iternova/nagios

- Crear un contenedor a partir de una imagen con un volumen

Docker con centreon

Docker file https://hub.docker.com/r/padelt/centreon/~/dockerfile/

Docker servicios web

Instalación

Dar de alta un usuario en mongo

```
use adim
db.auth("root", "root")
db.addUser("usuario", "pass") //Mirar los datos en el proyecto en core/controller/database/setting/database_mongodb.ini
```

Acciones varias para poner todo en marcha

1. ==========
2. Instalar mysql-client y mysql-server
3. modificar el document root a /var/www sed "-s/orgen/destiono/g"
4. memcache para php5 (creo que el cliente)
5. Al fallar cosas, Jorge me comenta de instalar las siguientes cosas
6. De memcache, reocrdad que se instalan dos librerías para el cliente. Además de lo que hice pongo apt-get install php-pear php5-memcache
7. instalo php5-mysqlnd
8. Coherencia con usuarios y bases de datos
9. Borrar todas las colecciones de configuration
10. rebortar memcached para que lea de la base de datos y no de memcached
11. corregir y acutalizar el nombre de la base de datos de web.ini
12. crear el usuario de mongo que está en web ini
13. crear el usuario de mysql dándole los permisos correspondientes
14. GRANT ALL PRIVILEGES on smartroads_core.* to 'smartroads_core'@'localhost' identified by '73cc8ea241';
15. ahora hay que dar permisos a las carpetas según lo está pidiendo

---

Usuario:Manue

Se ha guardado tu edición. ×

De WikiEducator
< Usuario:ManuelRomero

Contenido

Acciones básicas

Arranca un terminal de contenedor nuevo para probar algo a partir de la imagen ubuntu:14.04

```
sudo docker run --name dwes_cordoba -ti ubuntu:14.04
```

- Con esto ya tengo un contenedor creado, llamado dwes_cordoba para probar cosas

Una vez que cierre el contenedor lo puedo volver a abrir cuando quiera

```
{22:56}~ ➲ sudo docker start  dwes_cordoba
dwes_cordoba
{22:56}~ ➲ sudo docker exec -ti dwes_cordoba /bin/bash
root@496b9ba15066:/#
```

Modificando variables de entorno tipo PATH

```
acceder al fichero /etc/enviroment
```

Creando una imagen a partir de un fichero dockerfile ubicado en el directorio actual

```
sudo docker build -t iternova/web .
```

- Accion

```
Crea una imagen llamada iternova de tag web con el contenido del dockerfile que hay en el mismo directorio
```

- Restricciones

```
El nombre/tag no puede tener mayúsculas
```

Poniendo un volumen

```
El volumen se especifica para cada contenedor
 sudo docker run --name iternova -v /var/www/iternova:/var/www -dti iternova/web
```

descripción

```
arranca un contenedor llamado iternova
creando un volumen /var/www/iternova en el contenedor con /var/www que está en el host
```

-dti (background terminal e interactivo) de una imagen llamada iternova

Arrancando acciones (servicios activos) en el contenedor

Qué es docker

Definición

Docker es un proyecto Open-Source que automtiza ***el despliegue de aplicaciones dentro de un container de software***

- No es una máquina virtual.
- Con docker se generan "containers" independientes para ejecutarse en una simple instancia de Linux, por lo que para tener diferentes configuraciones de entornos de desarrollo, no necestiamos tener diferentes máquinas virtuales. Un solo sistema operativo para n entornos o configuraciones diferentes de entornos de desarrollo basados en containers.

Ojo, qué no es docker

Qué si que hace docker

Instalando docker

- Instalando la última version de docker (acutalmente 1.9 (Versión estable)

```
wget -qO- https://get.docker.com/ | sh
```

- Visualizando la versión

```
root@ubuntu1404:~# docker -v
                   Docker version 1.9.1, build a34a1d5
```

- Instalando versión más reciente (experimental)
  - Añadiendo la clave pública/privada de docker

```
apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 36A1D7869245C8950F966E92D8576A8BA88D21E9
```

- Añadimos el repositorio de docker en nuestro source.list

```
root@ubuntu1404:~# sh -c "echo deb https://get.docker.com/ubuntu docker main >> /etc/apt/sources.list.d/docker.
```

Crear un contenedor con volumenes a partir de una imagen

sudo docker run -t -i --name iternova5 -v /home/manolo/iternova:/local iternova-nagios-centreon-git /bin/bash

Ejecutar docker sin sudo

https://datafull.co/p/como-puedo-usar-docker-sin-sudo
https://docs.docker.com/engine/security/security/

Dockfile

- https://docs.docker.com/engine/reference/builder/
- Sacado de esta página
- Es un fichero de texto que me permite crear una imagen a partir de otra, y añadir una serie de comandos para instalar a esa imagen aplicaciones, servicios, librerías diversas con el objetivo de construirme una imagen para lanzar un contenedor personalizado a mis necesidades. Posteriormente puedo crear una imagen con ese contenedor.
- Para ejecutar la máquina ejecutamos

```
docker build .
```

Estructura del fichero

```
# comentarios
INSTRUCCIONES argumentos
```

En un fichero dockerfile tendremos dos tipos de líneas

1. comentarios (líneas que empiezan por **#**)
2. Sentencias líneas que tiene una instrucción con una serie de argumentos para ejecutarlas
    1. Por se recomienda poner las instrucciones en mayúsculas

Instrucciones

FROM

```
FROM <imagen>
FROM <imagen>:<tag>
FROM <imagen>@<digest>
```

- Es la primera instrucción
- Especifica la imagen base de la que parte la instalación que queremos hacer
- En esta página puedes encontrar imágenes base de las cuales partir (hay miles)
- En el FROM se puede especificar de forma opción el tag o digest (sirve para especificar la imagen, como por ejemplo la versión).
- Lo primero que tenemos que especificar es la imagen de la que partimos ***instruccion FROM***

Ejemplo

```
;imagen en base a ubuntu 14.04
FROM ubuntu:14.04
```

```
;imagen en base a debian
FROM debian:stable
```

MAINTAINER

```
MAINTAINER <name>
```

- Especifica el autor de la imagen y su correo

MAINTAINER Manuel Romero <ManuelRomeroMiguel@gmail.com>

RUN

```
RUN <command>
RUN ["ejecutable", "parametro1", "parametro2"]
```

- Ejecuta cualquier comando en la imagen que estamos usando. El resultado de la ejecución se puede tomar para el siguiente paso en el dockfile, por ejemplo si ejecuto la instalación de apache, en el siguiente comando puedo contar que apache ya está instalado
- Es un comando muy utilizado.
- Si usamos apt-get install especificaremos el flag -y para que no haga falta que sea interactivo y tome la decisión por defecto

- Añadimos a nuestro fichero las siguientes librerías

```
#Ahora empezamos a instalar todos los paquetes necesarios

#vim por ser un editor que me gusta
RUN apt-get intall -y vim


#gearman => Un demonio para ejecutar tareas pesadas de la aplicación web en background

RUN apt-get install -y gearman gearman-job-server gearman-tools libgearman-dev



# APC: Advanced PHP Cache
RUN apt-get install -y  php-apc


RUN echo "apc.enable_cli=1" >> /etc/php5/mods-available/apcu.ini
RUN echo "apc.enable=1" >> /etc/php5/mods-available/apcu.ini
RUN echo "extension=apcu.so" >> /etc/php5/mods-available/apcu.ini
```

Rehaciendo todo

- Creo un contenedor de debian

```
sudo docker run -i -t debian:jessie /bin/bash
```

- Instalo apache

```
apt-get update
apt-get install apache2
```

- Información de docker

```
sudo docker info
```

- Hacer un commit, para fijar una imagen

sudo docker **container_id** nombre_imagen_creado

- Arrancar un contenedor a partir de un commit

Comandos para recordar

Ver los contenedores arrancadso

```
docker ps
```

Ver todos los contenedores del sistema

```
docker ps -a
```

Arrancar un contenedor parado

- Primero localizar su nombre o su id (docker ps -a)

```
sudo docker start nombreContenedor
```

- Ahora cargamos bash para interactuar con ese comando

```
sudo docker exec -t -i nombreContenedor /bin/bash
```

Copiar ficheros a un contenedor

Referencias en la web

- https://docs.docker.com/userguide/
- https://www.youtube.com/playlist?list=PLfW3im2fiA7W9F4DbjmRDIZgAHsea20ON
- http://www.muylinux.com/2016/04/19/tutorial-docker
- https://github.com/brunocascio/docker-espanol (Muy bueno y práctico)

Usuario:ManuelRomero/proyecto/proyectoIternova/git

De WikiEducator
< Usuario:ManuelRomero | proyecto

https://wiki.iternova.net/doku.php?id=development:servers:howto_gitlab&s[]=mysql&s[]=smartroads

```
git branch //Para ver qué rama hay

git add * //para añadir los ficheros
git commit -m "Mensaje" //para hacer el comit con menaje
git push origin manuel //para subirlo al servidor a mi branch


gitg #pequeño programa que hay que instalar para ver de forma gráfica las ramas
```

Usuario:ManuelRomero/vagrant

De WikiEducator
< Usuario:ManuelRomero

Vagrant

- Es una herramienta para crear un entorno de virtualización para desarrollar
- Estos entornos serán portables, ligeros y reproducibles
- Está basado en un boxes o cajas que ya están previamente preinstaladas, por lo que no hay que crear una máquina virtual desde cero como con virtualbox solo.
- Estos ficheros de boxes, una vez descargados se guardan en ~/.vagrant.d/boxes
- La configuración de vagrant está basado en un fichero llamado ***vagrantfile*** . Solo un vagrantfile por proyecto.
- Existe una herramienta llamada puphpep. Con ella podemos crear de manera gráfica el fichero vagrantfile.
- Visitar la página https://puphpep.com
-
- Instalamos desde la página oficial el vagrant
- Tenemos instalado virtualbox en la pagina
- http://www.slideshare.net/vallekano85/virtualizacin-y-provisionamiento-entornos-de-desarrollo-con-vagrant-y-puppet?ref=http://jarroba.com/como-crear-entornos-de-desarrollo-con-vagrant-y-puppet/

Acciones

Instalar vagrant instalar virtual vox Cargar un box con vagrant

```
vagrant box add  puphpet/debian75-x64
```

Lista de boxes disponibles

```
https://atlas.hashicorp.com/boxes/search
```

Creamos un fichero Vagrantinit

```
vagrant init
```

Editamos el fichero de vagrantFile y añadimos Vagrant.configure("2") do |config|

```
 config.vm.box = "puphpet/debian75-x64"
```

o

```
vagrant up
vagrant ssh
```

Notas sobre vagrant

- La carpeta donde lance vagrant es compartida por la mv y la máquina real

---