Práctica 2
PHP + MySQL

Programación Web

3º Grado en Ingeniería Informática - Universidad de Córdoba

Jose María Moyano Murillo

jmoyano@uco.es

1. Introducción

Práctica 2 - PHP + MySQL

Introducción
Problema

- En la anterior práctica, se necesitaba un nuevo fichero HTML para cada libro de la biblioteca, y había que modificar los listados cada vez que un libro se añadía, modificaba o eliminaba.
- La web en solo HTML es muy estática y difícil de mantener.
- El uso de PHP y MySQL permite hacer la web mucho más flexible y obtener los datos de una base de datos.

2. PHP

Práctica 2 - PHP + MySQL

PHP
Introducción

- Con PHP es simple incluir contenido dinámico a páginas web.
- Es decir, el contenido de la página puede ser diferente cada vez que el navegador pide una página.
- Dándole a un fichero la extensión *.php*, se considera directamente como un fichero de este lenguaje de scripting.

```
sudo apt-get install php7.0
sudo apt-get install php7.0-mysql
```

PHP
IDE

- Hay multitud de editores o IDEs para desarrollar en PHP.
- PhpStorm es un IDE muy usado para el desarrollo de PHP que incluye una licencia gratuita para estudiantes.

https://www.jetbrains.com/phpstorm/

PHP
Servidor

- PHP incluye un servidor propio.
- El servidor de PHP se arranca con la siguiente instrucción por consola:

```
php -S localhost:8000 -t directorio
```

- El servidor busca automáticamente un fichero *index.php* o *index.html* en el directorio indicado.
- Acceder desde el navegador como:

```
localhost:8000
localhost:8000/index.php
```

PHP
Hola mundo

- Dentro del fichero *.php*, el código en PHP debe ir entre los símbolos `<?php … ?>`.
- En PHP, el programa *Hola mundo* sería:

```
<?php
   echo "Hola mundo!"
?>
```

PHP
Etiquetas <?php … ?>

- Todo el código que no se encuentre dentro de las etiquetas <?php … ?> se envía al cliente como código HTML.
- Por ejemplo, en el siguiente código:

```
<?php
   echo "Today is " . date("l") . ". ";
?>
Here's the latest news.
```

- La etiqueta <?php permite a PHP interpretar el código incluido hasta la etiqueta ?>.
- La última línea simplemente se muestra en el navegador como si estuviese en un fichero HTML.

```
Today is Thursday. Here's the latest news.
```

PHP
Comentarios

- PHP permite incluir tanto comentarios de una línea como de varias líneas.

```
//Comentario de una linea

/*
 * Comentario
 * de varias
 * lineas
 */
```

PHP
Sintaxis básica

- Cada sentencia ha de terminar con punto y coma “;”.

```php
$x += 10;
```

- El símbolo $ se ha de colocar delante de **todas** las variables.

```php
$contador = 1;
$real = 1.5;
$cadena = "Hola";
$cadena2 = 'mundo!';
$array = array("Uno", "Dos", "Tres");
```

- No hay que indicar el tipo de la variable.

PHP
Arrays

- Los arrays se crean con el constructor `array();` incluyendo dentro los valores separados por comas.

```php
$equipo = array('Jose', 'Pablo', 'Lucia', 'Mario', 'Paula');
```

- Para acceder a un elemento del array, se hace con corchetes.

```php
echo $equipo[2]; //Muestra el nombre Lucia
```

PHP
Arrays

- Anidando arrays, PHP permite crear arrays de dos o más dimensiones.

```php
$matrix = array(
    array('c00', 'c01', 'c02'),
    array('c10', 'c11', 'c12'),
    array('c20', 'c21', 'c22')
);
```

- Igual que antes, se accede con corchetes a las celdas.

```php
echo $matrix[1][2]; //Muestra la cadena 'c12'
```

PHP
Arrays

- También se pueden crear arrays sin definirlos previamente.
- Cada vez que se asigna un elemento al array, se incluye en el primer hueco libre.

```php
$comprar[] = "Leche";
$comprar[] = "Huevos";
$comprar[] = "Azucar";
$comprar[] = "Pan";

print_r($comprar);
```

```
Array
(
[0] => Leche
[1] => Huevos
[2] => Azucar
[3] => Pan
)
```

- Se puede hacer lo mismo especificando las posiciones concretas

```php
$comprar[0] = "Leche";
$comprar[1] = "Huevos";
$comprar[2] = "Azucar";
$comprar[3] = "Pan";
```

PHP
Arrays

- El uso de índices numéricos en los arrays puede ser útil, pero puede requerir un trabajo extra recordando qué índice corresponde a cada producto.
- Los arrays asociativos permiten referenciar los elementos por nombre y no por índice numérico.
- Se puede crear con constructor o sin él.

```php
$comprar = array(
    'leche' => "Covap",
    'huevos' => "Coren",
    'azucar' => "Azucarera",
    'pan' => "Bimbo");

echo $comprar['leche'];
```

```php
$comprar['leche'] = "Covap";
$comprar['huevos'] = "Coren";
$comprar['azucar'] = "Azucarera";
$comprar['pan'] = "Bimbo";
```

PHP
Arrays - Funciones

- Saber si una variable es un array

```php
is_array($comprar);
```

- Conocer el número de elementos que tiene un array

```php
count($comprar); //Solo tiene en cuenta el primer nivel del array
count($multiArray, 1); //Tiene en cuenta recursivamente todos los niveles
                       (arrays multi-dimensionales)
```

- Ordenar un array

```php
sort($comprar, SORT_STRING);
sort($array, SORT_NUMERIC);
rsort($comprar, SORT_STRING);
```

  - `sort` funciona directamente sobre el array.

PHP
Arrays - Funciones

- Desordenar un array (actúa directamente sobre el array)

```php
shuffle($numeros);
```

- Separar una cadena de caracteres por un delimitador

```php
$temp = explode(' ', "Esto es una frase con siete palabras");
print_r($temp);
```

```
Array
(
[0] => Esto
[1] => es
[2] => una
[3] => frase
[4] => con
[5] => siete
[6] => palabras
)
```

PHP
Operadores

Operadores aritméticos		
Operador	**Descripción**	**Ejemplo**
+	Suma	`$i+1;`
-	Resta	`$i-6;`
*	Multiplicación	`$i*10;`
/	División	`$i/2;`
%	Módulo	`$i%5;`
++	Incremento	`$i++;`
--	Decremento	`$i--;`

PHP
Operadores

Operadores de asignación		
Operador	**Ejemplo**	**Equivalente a**
=	`$j = 15`	`$j = 15`
+=	`$j += 10`	`$j = $j + 10`
-=	`$j -= 5`	`$j = $j - 5`
*=	`$j *= 3`	`$j = $j * 3`
.=	`$j .= $k`	`$j = $j . $k`
…		

PHP
Operadores

Operadores de comparación		
Operador	**Descripción**	**Ejemplo**
==	Igual	`$i == 3`
!=	Distinto	`$i != 5`
>	Mayor que	`$i > 1`
<	Menor que	`$i < 10`
>=	Mayor o igual que	`$i >= 3`
<=	Menor o igual que	`$i <= 8`

PHP
Operadores

- PHP automáticamente convierte el tipo de las variables antes de compararlas con el operador ==.
- El operador === previene de la conversión de tipo.

```php
$a = "1000";
$b = "+1000";
if ($a == $b) echo "1"; //Se imprime
if ($a === $b) echo "2"; //No se imprime
```

PHP
Operadores

Operadores lógicos		
Operador	**Descripción**	**Ejemplo**
&&	*And*	`$i == 3 && $j == 1`
and	*Low-precedence and*	`$i == 3 and $j == 1`
||	*Or*	`$i < 5 || $j > 10`
or	*Low-precedence or*	`$i < 5 or $j > 10`
!	*Not*	`!($i == $j)`
xor	*Exclusive or*	`$i xor $j`

PHP
Cadenas. Concatenación

- Se pueden concatenar cadenas con el operador punto `.`.

```php
echo "Tiene " . $msgs . " mensajes.";
```

- Similar al operador `+=`, se puede concatenar una cadena con otra con el operador `.=`.

```php
$cadena1 .= $cadena2
```

PHP
Cadenas. Tipos

- PHP distingue entre dos tipos de cadenas, dependiendo de las comillas utilizadas.
- Utilizando comillas simples, se asigna una cadena literal, conservando el contenido exacto de la cadena.

```php
$info = 'Preceder variables con el símbolo $ como: $var';
```

- Usando comillas dobles, PHP evaluará `$var` como una variable, incluyendo su valor.

```php
echo "Esta semana $count personas han visto tu perfil";
```

PHP
Cadenas. Caracteres de escape

- Algunas veces las cadenas pueden tener caracteres que induzcan a errores.

```
$text = 'My spelling's atroshus'; // Sintaxis erronea
```

- La solución es colocar una barra invertida justo antes del carácter problemático

```
$text = 'My spelling\'s still atroshus';
```

PHP
Comandos de múltiples líneas

- Algunas veces necesitamos mostrar mucho texto en PHP, que haciéndolo usando varias sentencias `echo` sería muy tedioso.
- Para ello, PHP incluye dos soluciones.
  1. Poner varias líneas entre las comillas en la sentencia `echo`.

```
echo "Así estaremos mostrando
una salida con varias líneas.

Fin."
```

PHP
Comandos de múltiples líneas

- Algunas veces necesitamos mostrar mucho texto en PHP, que haciéndolo usando varias sentencias `echo` sería muy tedioso.
- Para ello, PHP incluye dos soluciones.
  2. Utilizar el operador `<<<`, que preserva los saltos de línea y otros espacios.

```
echo <<<_END
        Esta es otra manera de imprimir
        varias líneas conservando los espacios.

        El último _END debe aparecer justo al principio de línea
        y sin contener nada más después, ni si quiera espacios ni comentarios.
_END;
```

PHP
Variables globales y estáticas

- Las variables globales son accesibles desde todo el código.
- Para declarar una variable como global se utiliza la palabra `global`.

```php
global $is_logged_in;
```

- Las variables estáticas mantienen su valor durante la ejecución.
- Se declara una variable estática con la palabra `static`.

```php
function test()
{
    static $count = 0; //Solo se puede inicializar con valores predefinidos
    echo $count;
    $count++;
}
```

PHP

Condicionales. `if ... elseif ... else`

```php
if ($balance < 100)
{
    $dinero = 1000;
    $balance += $dinero;
}
elseif ($balance > 200)
{
    $ahorros += 100;
    $balance -= 100;
}
else
{
    $ahorros += 50;
    $balance -= 50;
}
```

PHP
Condicionales. switch

```php
switch ($pagina)
{
    case "Principal":
        echo "Has selecionado Principal";
        break;
    case "Noticias":
        echo "Has selecionado Noticias";
        break;
    case "Login":
        echo "Has selecionado Login";
        break;
    default:
        echo "Selección no reconocida";
        break;
}
```

PHP
Bucles. `while`

```php
$combustible = 10;
while ($combustible > 1)
{
    // Continuar conduciendo ...
    echo "Hay suficiente cobustible";
}
```

PHP
Bucles. do ... while

```php
$count = 1;
do{
    echo "$count por 12 es " . $count * 12 . "<br>";
}while (++$count <= 12);
```

PHP
Bucles. for

```php
for ($count = 1 ; $count <= 12 ; ++$count){
    echo "$count por 12 es " . $count * 12 . "<br>";
}
```

PHP

Bucles. `foreach ... as`

```php
$comprar = array("Huevos", "Leche", "Azucar", "Pan");

foreach($comprar as $item)
{
    echo "$item<br>";
}
```

PHP
Funciones

```php
function function_name([parameter [, ...]])
{
    // Statements
}
```

- Los nombres de funciones son *case-insensitive*.
- Con `return` podemos devolver valores en la función.

```php
return $n1;

return array($n1, $n2, $n3);
```

PHP
Incluir y requerir ficheros

- Puede que necesitemos crear librerías o ficheros con funciones que necesitaremos después.
- La sentencia `include` dice a PHP que cargue todo el contenido de un fichero en concreto. Es lo mismo que si insertásemos todo el código del fichero en ese punto concreto.

```php
include "library.php";
```

- Cada vez que se usa `include`, se incluye el fichero completo, lo que puede llevar a errores si se incluye más de una vez. Usando `include_once` solamente se incluirá la primera vez.

```php
include_once "library.php";
```

PHP
Incluir y requerir ficheros

- Tanto `include` como `include_once` intentan incluir el fichero, pero si no lo encuentra, la ejecución continuará sin problema.
- Si es esencial incluir el fichero, es necesario utilizar `require` o `require_once`.

```php
require_once "library.php";
```

PHP
Incluir y requerir ficheros

- De esta manera, podemos crear **plantillas** de partes del sitio web.
- Por ejemplo, un fichero *header.php* que incluya el código de la cabecera de la web o un *footer.php* que incluya el pie de página.
- Después, en cada fichero *php* en el que lo queramos incluir bastará con utilizar una de las sentencias anteriores.
- Asi, en caso de querer modificarlo, únicamente habrá que modificar un fichero y no todos donde esté incluido.

```php
<?php
/* Includes the footer */
include('footer.php');
?>

</body>
```

PHP
Clases

```php
class User
{
    public $name, $password;

    function save_user()
    {
        //Code here
    }
}

$user = new User;
print_r($user); //Imprime información de un objeto de forma legible

$user->name = "Jose";
echo $user->name;
$user->save_user();
```

PHP
Clases

- Al pasar un objeto como parámetro, se pasa por referencia.
- Por lo que si se modifica dentro de una función, se modificará el objeto original también.
- Para evitar problemas, se puede utilizar el operador `clone`, que crea una copia del objeto.

```php
$u2 = clone $u;
```

PHP
Clases

- El constructor de una clase se crea con la función `__construct`.
- Aunque también se puede crear con una función con el mismo nombre de la clase, su uso no está recomendado.

```php
function __construct($name, $password)
{
    $this->name = $name;
    $this->password = $password;
}
```

PHP
Herencia

- PHP permite crear clases que hereden de otras con la sentencia `extends` en la definición de la clase.

```php
class Subscriber extends User
{
    public $phone, $email;
    function display()
    {
        echo "Nombre: " . $this->name . "<br>";
        echo "Contraseña: " . $this->password . "<br>";
        echo "Telefono: " . $this->phone . "<br>";
        echo "E-mail: " . $this->email;
    }
}
```

PHP
Herencia

- Al crear una función en la clase hija con el nombre de una función de la clase padre, esta función sobrescribirá a la del padre.
- Se permite acceder a la función de la clase padre con la sentencia `parent`.

```
parent::display();
```

- Al crear el constructor de una clase hija, asegurar que se llama al constructor de la clase padre.

```
function __construct()
{
    parent::__construct(); // Llamar primero al constructor de la clase padre
    //Mas codigo del constructor aqui
}
```

3. MySQL

Práctica 2 - PHP + MySQL

MySQL
Acceso por consola

- Para acceder por consola a `mysql` se hace de la siguiente manera

```
mysql –u user -p
mysql –u root
```

```
sudo apt-get mysql-server
sudo apt-get mysql-client
```

MySQL
Acceso UCO

- Para acceder a mysql desde los ordenadores de la Universidad de Córdoba hay que seguir las instrucciones del siguiente enlace:

  http://oraclepr.uco.es/abd/

- Acceder a *My Base de Datos* en el menú izquierdo y crear la base de datos.
  - El nombre de la base de datos creada será el nombre de usuario
  - El usuario y contraseña de mysql son los mismos que de la UCO
  - Servidor: oraclepr.uco.es

MySQL
Comandos básicos

- Mostrar bases de datos

```
SHOW DATABASES;
```

- Crear base de datos

```
CREATE DATABASE library;
```

- Usar base de datos

```
USE library;
```

MySQL
Crear usuarios

- Se crean usuarios con CREATE USER.

```
CREATE USER 'username'@'hostname' IDENTIFIED BY 'password';
CREATE USER 'jose'@'localhost' IDENTIFIED BY 'josePWD';
```

- Se puede dar privilegios a los usuarios con GRANT:

```
GRANT [privilege] ON [database].[table] TO 'username'@'hostname';
GRANT ALL PRIVILEGES ON library.* TO 'jose'@'localhost';
GRANT CREATE ON *.* TO 'jose'@'localhost';
GRANT SELECT ON *.* TO 'jose'@'localhost';
```

MySQL
Crear tabla

- Para crear una tabla se hace con `CREATE TABLE`.

```
CREATE TABLE books(
  /* Define the columns */
  title VARCHAR(256) NOT NULL,
  author VARCHAR(256) NOT NULL,
  category VARCHAR(128),
  isbn VARCHAR(17) UNIQUE NOT NULL,
  id INT UNSIGNED NOT NULL AUTO_INCREMENT,

  PRIMARY KEY(id)
);
```

- Mostrar descripción de tabla

```
DESCRIBE books;
```

MySQL
Tipos de datos

Tipo de dato	Bytes usados	Descripción
CHAR(*n*)	Exactamente *n* (<=255)	Cadena de caracteres
VARCHAR(*n*)	Hasta *n* (<=65535)	Cadena de caracteres
TINYTEXT(*n*)	Hasta *n* (<=255)	Texto
TEXT(*n*)	Hasta *n* (<= 65535)	Texto
MEDIUMTEXT(*n*)	Hasta *n* (<= 1.67e+7)	Texto
LONGTEXT(*n*)	Hasta *n* (<= 4.29e+9)	Texto

MySQL
Tipos de datos

Tipo de dato	Bytes usados	Valor mínimo		Valor máximo	
		Signed	*Unsigned*	*Signed*	*Unsigned*
TINYINT	1	-128	0	127	255
SMALLINT	2	-32768	0	32767	65535
MEDIUMINT	3	-8.38e+6	0	8.38e+6	1.67e+7
INT / INTEGER	4	-2.15e+9	0	2.15e+9	4.29e+9
BIGINT	8	-9.22e+18	0	9.22e+18	1.84e+18
FLOAT	4	-3.40e+38	-	3.40e+38	-
DOUBLE / REAL	8	-1.80e+308	-	1.80e+308	-

MySQL
Insertar datos en una tabla

- Se insertan datos en una tabla con `INSERT INTO`.

```
INSERT INTO books (title, author, category, isbn)
VALUES(
  'Don Quijote de la Mancha',
  'Miguel de Cervantes',
  'Novela',
  '978-8-46-632040-5',
);
```

MySQL
Otras utilidades

- Eliminar una tabla

```
DROP TABLE books;
```

- Renombrar una tabla

```
ALTER TABLE books RENAME libros;
```

- Añadir columnas a una tabla

```
ALTER TABLE books ADD pages SMALLINT UNSIGNED;
```

- Borrar columnas

```
ALTER TABLE classics DROP pages;
```

MySQL
Consultas

- Obtener la tabla *books* completa.

```
SELECT * FROM books;
```

- Obtener el título e isbn de los libros en la tabla *books*

```
SELECT title, isbn FROM books;
```

- Obtener el recuento de libros

```
SELECT COUNT(*) FROM books;
```

- Obtener los distintos autores

```
SELECT DISTINCT(author) FROM books;
```

MySQL
Consultas

- Obtener los libros de un determinado autor

```
SELECT author, title FROM books WHERE author='Mark Twain';
```

- Obtener todos los libros cuyo autor empiece por el nombre *Mark*

```
SELECT author, title FROM books WHERE author LIKE "Mark%";
```

- Obtener tan solo los tres primeros libros de la tabla

```
SELECT author, title FROM books LIMIT 3;
```

- Obtener dos libros excluyendo los cinco primeros

```
SELECT author, title FROM books LIMIT 5, 2;
```

MySQL
Consultas

- Obtener los libros ordenados por autor y título

```
SELECT author, title FROM books ORDER BY author, title;
```

- Obtener el número de libros de cada autor

```
SELECT author, COUNT(title) FROM books GROUP BY author;
```

- Crear un alias de una columna en la consulta

```
SELECT author, COUNT(title) AS c FROM books GROUP BY author;
```

- Obtener todos los libros de varios autores

```
SELECT author, title FROM books WHERE author='Mark Twain' OR
author='Miguel de Cervantes';
```

MySQL
Consultas

- Eliminar un libro

```
DELETE FROM books WHERE title='Don Quijote de la Mancha';
```

- Eliminar todos los libros de un autor

```
DELETE FROM books WHERE author='Mark Twain';
```

- Modificar el nombre de un autor en todos sus libros

```
UPDATE books SET author='M. Twain' WHERE author='Mark Twain';
```

4. Acceder a MySQL con PHP

Práctica 2 - PHP + MySQL

Acceder a MySQL con PHP
Problema HTML

- Hay fragmentos de código idénticos repartidos por todos los ficheros *html*.
- Hay muchos ficheros *html* con la misma estructura pero distinta información.
- El mantenimiento del sitio web es muy difícil y tedioso (añadir o modificar entradas supone modificar varios ficheros).

Acceder a MySQL con PHP

Problema HTML

index.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <!-- ... -->
</head>

<body>
    <header> <!-- ... --> </header>

    <table>
        <tr>
            <th>Libro</th>
            <th>Autor</th>
            <!-- ... -->
        </tr>
        <tr>
            <td>El Principito</td>
            <td>Antoine de Saint-Exupéry</td>
            <!-- ... -->
        </tr>
        <!-- ... -->
    </table>

    <footer> <!-- ... --> </footer>

</body>
</html>
```

principito.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <!-- ... -->
</head>

<body>
    <header> <!-- ... --> </header>

    <h3 align="center">El Principito</h3>

    <ul class="listInfo">
        <li><b>Autor:</b> Antoine de Saint-Exupéry</li>
        <li><b>Categoría:</b> Novela</li>
        <li><b>ISBN:</b> 978-8-49-838149-8</li>
        <!-- ... -->
    </ul>

    <footer> <!-- ... --> </footer>

</body>
</html>
```

index.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <!-- ... -->
</head>

<body>
    <header> <!-- ... --> </header>

    <table>
        <tr>
            <th>Libro</th>
            <th>Autor</th>
            <!-- ... -->
        </tr>
        <tr>
            <td>El Principito</td>
            <td>Antoine de Saint-Exupéry</td>
            <!-- ... -->
        </tr>
        <!-- ... -->
    </table>

    <footer> <!-- ... --> </footer>

</body>
</html>
```

index.php

```php
<!DOCTYPE html>
<html lang="es">

<head> <!-- ... --> </head>

<body>
<?php
include('header.php');
require_once('queryFunctionsLibrary.php');

$q = new LibraryQueries();

echo <<< END
<table>
    <tr>
        <th>Libro</th>
        <!-- ... -->
    </tr>
_END;

$books = $q->getAllBooks();
foreach($books as $book){
    echo <<<_END
    <tr>
        <td>$book[title]</td>
        <!-- ... -->
        <td><a href="/book.php?isbn=$book[isbn]">info</a></td>
    </tr>
_END;
}

echo "</table>";

include('footer.php');
?>
</body>
</html>
```

/book.php?isbn=…

header.php

```php
<header>
    <!-- ... -->
</header>
```

footer.php

```php
<footer>
    <!-- ... -->
</footer>
```

principito.html

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <!-- ... -->
</head>

<body>
    <header> <!-- ... --> </header>

    <h3 align="center">El Principito</h3>

    <ul class="listInfo">
        <li><b>Autor:</b> Antoine de Saint-Exupéry</li>
        <li><b>Categoría:</b> Novela</li>
        <li><b>ISBN:</b> 978-8-49-838149-8</li>
        <!-- ... -->
    </ul>

    <footer> <!-- ... --> </footer>

</body>
</html>
```

/principito.html
/sinsajo.html
...

book.php

```php
<!DOCTYPE html>
<html lang="es">

<?php
/* Get the isbn from the _GET array */
$isbn = $_GET['isbn'];

require_once('queryFunctionsLibrary.php');

$book = $q->getBook($isbn);

echo <<<_END
    <head> <!-- ... --> </head>
_END;
?>

<body>

<?php
include('header.php');

echo <<< END
    <h3>$book[title]</h3>
    <ul>
        <li><b>Autor:</b> $book[author]</li>
        <li><b>Categoría:</b> $book[category]</li>
        <li><b>ISBN:</b> $book[isbn]</li>
    </ul>
_END;
?>

</body>

</html>
```

/book.php?isbn=98-8-49-838149-8
/book.php?isbn=98-8-42-156823-1

Acceder a MySQL con PHP
Conectar con la base de datos

- Creamos una clase que conectará con la base de datos e incluirá las consultas a la misma.
- Primero incluimos el usuario y contraseña de MySQL y el objeto para conectarse a la base de datos.

```php
class LibraryQueries {
    //Usuario
    public $user = 'jose';

    //Contraseña
    public $password = 'josePWD';

    //Objeto PDO para conectar con la base de datos
    public $dbc;
```

Acceder a MySQL con PHP
Conectar con la base de datos

- A continuación, el constructor de la clase se encargará de llamar a la función que conecta con la base de datos y la tabla correspondiente, almacenando la conexión en el objeto *dbc*.

```php
/* Constructor. Llama a la función que conecta con la base de datos */
public function __construct(){
    //Conectar con la base de datos
    $this->dbc = $this->dbconnect();
}
```

Acceder a MySQL con PHP
Conectar con la base de datos

- La función encargada de conectar con la base de datos es la siguiente.

```php
/* Conectar con la base de datos y la tabla correspondiente*/
public function dbconnect() {
    $dbc = null;

    try {
        $dbc = new PDO('mysql:host=localhost; dbname=library', $this->user,
        $this->password, array(PDO::ATTR_PERSISTENT => true));

    } catch (PDOException $e) {
        return null;
    }

    return $dbc;
}
```

http://us3.php.net/manual/es/book.pdo.php

https://phpdelusions.net/pdo

Acceder a MySQL con PHP
Conectar con la base de datos

- En caso de estar usando mysql proporcionado por la UCO, tener en cuenta la sentencia de conexión con la base de datos debe ser la siguiente:

```
$dbc = new PDO('mysql:host=oraclepr.uco.es; dbname=usuarioUCO', ...
```

Acceder a MySQL con PHP
Consultas a la base de datos

- Dentro de la misma clase que hemos creado, podemos incluir funciones que accedan a la base de datos.

```php
public function getAllBooks(){
    $books = array();

    $sentence = $this->dbc->prepare("SELECT * FROM books");

    if ($sentence->execute()) {
        while ($row = $sentence->fetch()) {
            $books[] = $row;
        }
    }

    return $books;
}
```

Acceder a MySQL con PHP
Consultas a la base de datos

- De esta manera, crearemos **una función distinta por cada consulta** que necesitemos hacer para nuestro sitio web.
- Cada **función devolverá los datos** correspondientes (un array con la información de todos los libros, con la información de un libro o autor específico, etc.) y después el código **PHP mostrará** esta información **de una manera concreta**.
- Así **separamos el controlador** (acceso a la base de datos) **de la vista** (forma en la que se muestra) haciendo nuestro código lo más modular y comprensible posible.

Acceder a MySQL con PHP
Uso de las funciones de consulta

- En cada fichero de código PHP que necesitemos acceder a la base de datos, primero tendremos que requerir el fichero donde se definen las funciones de consultas.

```php
require_once('queryFunctionsLibrary.php');
```

- Creamos un objeto de la clase definida anteriormente.

```php
$q = new LibraryQueries();
```

- Aseguramos que se ha creado el objeto correctamente.

```php
if(empty($q->dbc)){
    //Error message
    die();
}
```

Acceder a MySQL con PHP
Uso de las funciones de consulta

- Para hacer una consulta a la base de datos, llamar a la función correspondiente.

```
$books = $q->getAllBooks();
```

- Mostrar los datos como convenga.

```
foreach($books as $book){
    echo <<<_END
    <tr align="center" class="list" >
        <td class="list" >$book[title]</td>
        <td class="list" >$book[author]</td>
        ...
    </tr>
_END;
}
```

Práctica 2 - Biblioteca PHP + MySQL

Programación Web

3º Grado en Ingeniería Informática - Universidad de Córdoba

Jose María Moyano Murillo

jmoyano@uco.es

Práctica 2 - Biblioteca
Objetivos

- Modificar la práctica anterior haciendo uso de PHP y MySQL.
- La información de los libros estará almacenada en una tabla en una base de datos de MySQL.
- Los datos de la base de datos se obtendrán mediante PHP.
- Así, el código HTML no deberá contener ninguna información acerca de los libros, autores, ect.

Práctica 2 - Biblioteca

Organización en ficheros

Práctica 2
PHP + MySQL

Programación Web

3º Grado en Ingeniería Informática - Universidad de Córdoba

Jose María Moyano Murillo

jmoyano@uco.es