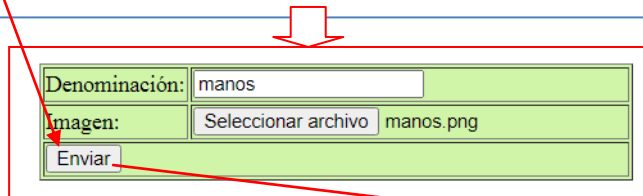


1. “Almacenar” y recuperar imágenes de una BD utilizando PHP

Ejemplo. Partimos de un formulario de entrada (*formulario.php*):

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<!--FORMULARIO -->
<br><br><br>
<table border=1 align="center" bgcolor="#D0F5A9">
<form action="datosimg.php" method="post" enctype="multipart/form-data"> <!--enctype necesario para trabajar con archivos -->
  <tr><td>Denominación:</td><td><input type="text" name="texto id"></td></tr>
  <tr><td>Imagen:</td><td><input type="file" name="imagen1" size="20"></td></tr>
  <tr><td colspan="2"><input type="submit" name="Subir imagen"></td></tr>
</form>
</table>
</body>
</html>
```



El fichero en el que recae el tratamiento de la imagen, podría ser el siguiente (*datosimg.php*):

```
?php

/* $_FILES es una variable superglobal que almacenará datos de la imagen recibida:
   name, almacena el nombre del fichero
   type, el tipo o extensión (se podrá limitar el tipo de archivos a subir)
   size, el tamaño (se podrá limitar el tamaño de los archivos a subir)
   tmp_name, el nombre del directorio temporal utilizado de forma previa al alojamiento definitivo del fichero
   que en este ejemplo será está en $dir
   error, si hay algún error*/
$nomf=$_FILES['imagen1']['name'];
$tipo=$_FILES['imagen1']['type'];
var_dump($tipo);
echo "Imagen recibida: ".$nomf;
echo "<br>";
$dir=$_SERVER['DOCUMENT_ROOT'].'/uploads/'; //será necesario crear el directorio uploads en la raíz del servidor web
var_dump($dir);
if($tipo=='image/jpeg' || $tipo=='image/png' || $tipo=='image/gif')
  move_uploaded_file($_FILES['imagen1']['tmp_name'],$dir.$nomf);
else
  echo "<br>No es una imagen";

include ("accesobd.php"); //acceder a la BD para insertar un registro

?>
```

string 'application/pdf' (length=15)
Imagen recibida: ActividadUT6_guiada.pdf
string 'C:/wamp/www/uploads/' (length=20)

No es una imagen

O
string 'image/png' (length=9)
Imagen recibida: respeto.png
string 'C:/wamp/www/uploads/' (length=20)



Una vez que las imágenes están en el servidor web, es posible incluirlas en un campo de una BD. Lo más normal es incluir la ruta de la imagen para evitar ocupar espacio en la BD. La BD utilizada será denominada *obras_arte*. Solo tendrá una tabla denominada *obras*. La tabla *obras*, tendrá dos campos (*deno*,*foto*), siendo necesario que el campo *foto* sea de tipo texto (VARCHAR) con una longitud adecuada para almacenar la ruta y el nombre de la foto.

```
mysql> desc obras;
```

Field	Type	Null	Key	Default	Extra
deno	varchar(10)	NO	PRI	NULL	
foto	varchar(50)	NO		NULL	

2 rows in set (0.03 sec)

Para “almacenar” la imagen en la BD será preciso el siguiente código (accesodb.php):

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<?php

$db_host="localhost";
$db_usuario="root";
$db_clave="";
$db_nombre="obras_arte";

$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexión#243n <br>";
    exit();
}
else
    echo "conexión#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontró#243n la BD");
$deno=$_POST['texto_id'];
$insertar='insert into obras (deno,foto) values (?,?)';
$stmt=mysqli_prepare($conexion,$insertar);
mysqli_stmt_bind_param($stmt,'ss',$deno,$nomf);
mysqli_stmt_execute($stmt);
mysqli_close($conexion);*/

/*$consulta="select foto from obras";
$resultados=mysqli_query($conexion,$consulta);
$fila=mysqli_fetch_array($resultados);
    $ruta_foto=$fila[0];
    echo $ruta_foto;
mysqli_close($conexion); */
?>
<!--<div>
    
</div> -->
</body>
</html>
```

Tras la ejecución del código del fichero accesodb.php, se insertarán una tupla con los datos capturados desde el formulario. En la BD se observará que la tupla ha sido insertada:

```
mysql> select * from obras;
```

deno	foto
manos	manos.png

1 row in set (0.00 sec)

Esta técnica es utilizada por muchos CMS; no almacenan las imágenes directamente en las BD sino sus rutas. Las imágenes en sí, quedan en el servidor web.

Ya tenemos la imagen “almacenada” en la BD. Sólo falta, recuperar la imagen de la BD para visualizarla en nuestra aplicación web.

Para ello, ejecutaremos de nuevo la aplicación, no hará falta que insertemos una nueva imagen desde el formulario pues lo necesario es ejecutar el código de accesodb.php que anteriormente viste comentado:

```
<?php

$db_host="localhost";
$db_usuario="root";
$db_clave="";
$db_nombre="obras_arte";

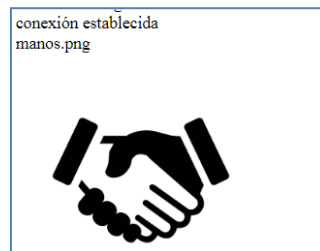
$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexi&#243n <br>";
    exit();
}
else
    echo "conexi&#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontr&#243 la BD");
/*$deno=$_POST['texto_id'];
$insertar='insert into obras (deno,foto) values (?,?)';
$stmt=mysqli_prepare($conexion,$insertar);
mysqli_stmt_bind_param($stmt,'ss',$deno,$nomf);
mysqli_stmt_execute($stmt);
mysqli_close($conexion);*/

$consulta="select foto from obras";
$resultados=mysqli_query($conexion,$consulta);
$fila=mysqli_fetch_array($resultados);
    $ruta_foto=$fila[0];
    echo $ruta_foto;
mysqli_close($conexion);
?>

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

Y se visualizará la imagen:



2. Utilizar el campo de tipo BLOB para subir archivos a una BD. Un campo BLOB permite almacenar un fichero. En este caso, partiremos de una nueva base de datos, obras_arte2, con una única tabla y la siguiente estructura:

```
mysql> desc obras;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| deno  | varchar(15) | NO   | PRI | NULL    |       |
| archivo | longblob | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.05 sec)
```

Reutilizaremos el código de la actividad 1 e insertaremos una tupla en esta tabla, teniendo en cuenta el nuevo campo de tipo longblob:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<!--FORMULARIO -->
<br><br><br>
<table border=1 align="center" bgcolor="#D0F5A9">
<form action="tratamiento.php" method="post" enctype="multipart/form-data"> <!--enctype necesario para trabajar con archivos -->
  <tr><td>Denominación:</td><td><input type="text" name="texto_id"></td></tr>
  <tr><td>Imagen: </td><td><input type="file" name="fichero" size="20"></td></tr>
  <tr><td colspan="2"><input type="submit" name="Subir archivo"></td></tr>
</form>
</table>
</body>
</html>
```

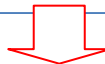


```
<?php

/* $_FILES es una variable superglobal que almacenará datos de la imagen recibida:
   name, almacena el nombre del fichero
   type, el tipo o extensión (se podrá limitar el tipo de archivos a subir)
   size, el tamaño (se podrá limitar el tamaño de los archivos a subir)
   tmp_name, el nombre del directorio temporal utilizado de forma previa al alojamiento definitivo del fichero
   que en este ejemplo será está en $dir
   error, si hay algún error*/
$nomf=$_FILES['fichero']['name'];
$t=$_FILES['fichero']['size'];
var_dump($t);
echo "Fichero subido: ".$nomf;
echo "<br>";
$dir=$_SERVER['DOCUMENT_ROOT'].'/uploads/'; //será necesario crear el directorio uploads en la raíz del servidor web
var_dump($dir);
if ($t<1000000)
  move_uploaded_file($_FILES['fichero']['tmp_name'],$dir.$nomf);
else
  echo "Demasiado grande";

include ("accesobd.php"); //acceder a la BD para insertar un registro

?>
```



```
<?php

$db_host="localhost";
$db_usuario="root";
$db_clave="";
$db_nombre="obras_arte2";

$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexi&#243n <br>";
    exit();
}
else
    echo "conexi&#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontr&#243 la BD");
$deno=$_POST['texto_id'];
$nombre=$dir.$nomf; //ruta del fichero cuyo contenido se insertará en la BD
$f=fopen($dir.$nomf,"r"); //el fichero tiene que ser abierto en modo lectura
var_dump($dir);
$contenido=fread($f,$t); //se lee su contenido
var_dump($contenido);
$contenido=addslashes($contenido); //para un escapado correcto de los datos
fclose($f);
$insertar="insert into obras (deno,archivo) value ('$deno','$contenido')"; //otra forma de realizar una consulta
$resultado=mysqli_query($conexion,$insertar);
if (mysqli_affected_rows($conexion)>0)
    echo "Se ha insertado correctamente la insercion";
else
    echo "Error; no se ha realizado la insercion";
mysqli_close($conexion);
?>
```

Ejecución: Se insertan los datos en el formulario, se sube el fichero a /uploads y después, se inserta la tupla en la tabla "obras".

Denominación:	tatus
Imagen:	Seleccionar archivo tatus.jpg
<input type="button" value="Enviar"/>	



Desde PHPMYADMIN

		deno	archivo
<input type="checkbox"/>	Editar	tatus	[BLOB - 275.6KB]

Para recuperar la información de la base de datos, realizaremos una consulta y visualizaremos la imagen del fichero:

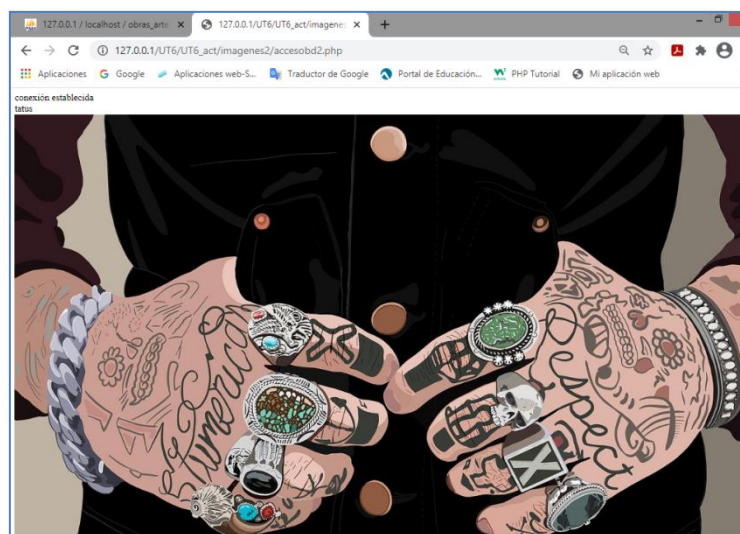
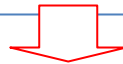
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<?php

$db_host="localhost";
$db_usuario="root";
$db_clave="";
$db_nombre="obras_arte2";

$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexi&#243n <br>";
    exit();
}
else
    echo "conexi&#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontr&#243 la BD");
$consulta="select * from obras where deno='tatus'";
$resultados=mysqli_query($conexion,$consulta);
$fila=mysqli_fetch_array($resultados);
$deno=$fila[0];
echo $deno;
$fichero=$fila[1];
mysqli_close($conexion);
//es necesario utilizar la función base64_encode ya que los datos del fichero están
//almacenados utilizando la codificación base64
echo "<img src='data:image/jpg; base64,\". base64_encode($fichero).\"'>";
?>
</body>
</html>
```



(fuente:<https://pixabay.com/es/illustrations/tatuaje-anillos-manos-respeto-2803482/>)

3. Otra forma de subir una imagen a una BD en este caso utilizando una consulta parametrizada y la función específica `mysqli_stmt_send_long_data`.

```
<?php
// Conexión y selección de la base de datos.
$db = mysqli_connect('localhost:3307','root','','obras_arte2');
if (!$db) {
    exit('Error de conexión.');
```

```
}
$sql = 'UPDATE obras SET foto = ? WHERE deno = ?';
$consulta = mysqli_prepare($db,$sql);
$ok = mysqli_stmt_bind_param($consulta,'bs' , $foto,$deno);
// Lectura del contenido del archivo imagen.
$ruta=$_SERVER['DOCUMENT_ROOT'].'/uploads/'.$tatus.jpg";
$foto = file_get_contents($ruta);
// Envío de los datos de la imagen.
$ok = mysqli_stmt_send_long_data($consulta,0,$foto);
// Ejecución de la consulta.
$deno="tatus";
$ok = mysqli_stmt_execute($consulta);
if ($ok) {
    echo "Actualización terminada con éxito <br>";
} else {
    echo "Error durante la actualización : ", mysqli_stmt_error($consulta);
}
// Desconexión.
$ok = mysqli_close($db);
?>
```

Para recuperar la imagen se puede utilizar el código de la actividad anterior.

4. Agrega un nuevo campo a la BD ciclo para incluir la foto de los alumnos. Recupera un alumno y muestra sus datos lo mejor que sepas.

Nota: Para crear o utilizar imágenes tal vez te resulten útiles...

- logomakr.com se pueden crear imágenes sencillas y gratuitas.
- unsplash.com se pueden descargar imágenes gratuitas.

5. Inserta las portadas de los libros utilizando un campo BLOB en la BD librería.

6. Registro y acceso a una BBDD. En este ejercicio, se utilizará una BD, denominada usuarios, con una única tabla con la siguiente estructura:

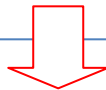
Field	Type	Null	Key	Default	Extra
id_usu	int(11)	NO	PRI	NULL	auto_increment
nombre	varchar(15)	NO		NULL	
clave	varchar(60)	NO		NULL	
rol	varchar(15)	NO		NULL	

Básicamente se trata de registrar usuarios en la BD, dando la opción de encriptar su clave. Para ello utilizaremos las funciones hash de PHP, donde el algoritmo de encriptación es *blogfish* (hay otros algoritmos de encriptación obsoletos como MD5, SHA-1,...debido a la facilidad de crakeo derivada de la información proporcionada por las tablas Rainbow (puedes consultar la información oficial de php para ampliar conocimientos: <https://www.php.net/manual/es/faq.passwords.php#faq.passwords.fasthash>). La función password_hash genera "la sal" de forma automática (también está en desuso la función crypt() la cual requiere de un proceso de generación de la sal, programado; en la documentación de php, por ello, se recomienda utilizar la librería nativa de php).

Para el proceso de registro, se utiliza el siguiente código (*registro.php*):


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<!--FORMULARIO -->
<br><br><br>
<table border=1 align="center" bgcolor="#D0F5A9">
<form action="tratamiento.php" method="post">
  <tr><td>Nombre usuario:</td><td><input type="text" name="usuario"></td></tr>
  <tr><td>Password:</td><td><input type="text" name="password"></td></tr>
  <tr><td>
    Elegir:<br>
    <input checked="" type="radio" name="c" value=1>Cifrar password<br>
    <input type="radio" name="c" value=2>No cifrar password<br>
  </td></tr>
  <tr><td colspan="2"><input type="submit" name="Registrar usuario"></td></tr>
</form>
</table>
</body>
</html>
```

Este es el tratamiento realizado (*tratamiento.php*):

```
<?php

$db_host="localhost:3308";
$db_usuario="root";
$db_clave="";
$db_nombre="usuarios";

$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexión#243n <br>";
    exit();
}
else
    echo "conexión#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontró#243n la BD");
$elegir=htmlentities(addslashes($_POST['c']));
$nombre=htmlentities(addslashes($_POST['usuario']));
$pass=htmlentities(addslashes($_POST['password']));
if ($elegir=='1')
{
    $password=password_hash($pass,PASSWORD_DEFAULT);
}
else
    $password=$pass;
$insertar="insert into usu (nombre,clave, rol) values ('$nombre','$password','usuario')"; //el campo clave será un varchar(60)
$resultado=mysqli_query($conexion,$insertar);
if (mysqli_affected_rows($conexion)>0)
    echo "Se ha insertado correctamente la inserción";
else
    echo "Error; no se ha realizado la inserción";
mysqli_close($conexion);
?>
```

Tras el registro de los usuarios, la BD, en concreto, la tabla "usu", quedaría así:

+ Opciones			id_usu	nombre	clave	rol
<input type="checkbox"/>	Editar	<input type="checkbox"/>	Copiar	<input type="checkbox"/>	Borrar	1 esther 12345 usuario
<input type="checkbox"/>	Editar	<input type="checkbox"/>	Copiar	<input type="checkbox"/>	Borrar	14 javier \$2y\$10\$XFX2Cc/pW1IFiY8xFNMniO97LJBvoQcgX7izm1s4zah... usuario
<input type="checkbox"/>	Editar	<input type="checkbox"/>	Copiar	<input type="checkbox"/>	Borrar	15 victor \$2y\$10\$liM/KBB8eCK9JP2QyCapJucv4EQzwmRK2alU3rRftC/... usuario

Una vez realizado el registro, se ejecuta el acceso (*acceso.php*):

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>

<!--FORMULARIO -->
<br><br><br>
<table border=1 align="center" bgcolor="#D0F5A9">
<form action="autorizar.php" method="post">
  <tr><td>Nombre usuario:</td><td><input type="text" name="usuario"></td></tr>
  <tr><td>Password:</td><td><input type="text" name="password"></td></tr>
  <tr><td colspan="2"><input type="submit" name="Registrar usuario"></td></tr>
</form>
</table>
</body>
</html>
```

Nombre usuario: esther

Password: 12345

Enviar

Acesso que será denegado

Nombre usuario: javier

Password: 12345

Enviar

Acesso que será autorizado

Y el tratamiento, está recogido en el siguiente código (*autorizar.php*):

```
<?php

$db_host="localhost:3308";
$db_usuario="root";
$db_clave="";
$db_nombre="usuarios";

$conexion=mysqli_connect($db_host,$db_usuario,$db_clave);
if (mysqli_connect_errno())
{
    echo "Fallo en la conexi&#243n <br>";
    exit();
}
else
    echo "conexi&#243n establecida <br>";

mysqli_select_db ($conexion,$db_nombre) or die ("No se encontr&#243 la BD");
$nombre=htmlspecialchars($_POST['usuario']);
$password=htmlspecialchars($_POST['password']);
$consultar="select * from usu where nombre='$nombre'; //el campo clave será un varchar(60)";
$resultado=mysqli_query($conexion,$consultar);
$comprobar=0; //controla los usuarios con clave no encriptada
$comprobar1=0; //controla los usuarios con clave encriptada
if (mysqli_affected_rows($conexion)>0)
{
    $comprobar=1;
    while ($fila=mysqli_fetch_array($resultado))
        if (password_verify($password, $fila['clave']))
            $comprobar1=1;
}
if ($comprobar==1 || $comprobar1==1)
    echo "Acceso autorizado".$fila['nombre'];
else
    echo "Acceso denegado. No está registrado";

mysqli_close($conexion);
?>
```

conexión establecida
Acceso denegado. No está registrado

conexión establecida
Acceso autorizado

7. Utilizando la POO, programar una conexión y realización de una consulta a una BD. Será para ello preciso, crear nuestras propias clases.

El código, podría ser el siguiente:

Partimos de un archivo *index.php* desde el cual se instancia un objeto asociado a una consulta:

```
<?php
echo "pasa";
require ("bd_class.php");
$objeto=new Consultas();
$alumnos=$objeto->GetDatos(); //$alumnos será un array
echo "<table border=1>";
echo "<tr><th>ID ALUMNO</th><th>NOMBRE</th><th>EDAD</th><th>CURSO</th></tr>";
foreach ($alumnos as $al)
{
    echo "<tr><td>".$al['id_al']."</td>";
    echo "<td>".$al['nombre']."</td>";
    echo "<td>".$al['edad']."</td>";
    echo "<td>".$al['codigo']."</td></tr>"; //en la BD desde la que probé, no tenía datos
}
echo "</table>";
?>
```

En el siguiente archivo, se detalla la codificación de las clases definidas, para este pequeño ejemplo de POO para acceder a una BD:

```
<?php
//ejercicio previo al MVC
require("datos_config.php");
class Conexiones {

    protected $bd_conect; //es un objeto, el mismo que hemos manejado en ejercicios anteriores para crear una conexión utilizando
    //la extensión mysqli con POO

    public function __construct()
    {
        $this->bd_conect=new mysqli(BD_H,BD_U,BD_PW,BD_NOM);

        if ($this->bd_conect->connect_errno)
            echo "No se ha podido establecer la conexión";
        else
            echo "Conexión establecida";

        $this->bd_conect->set_charset(BD_C); //para establecer el código UTF-8
    }

}

class Consultas extends Conexiones {

    public function __construct() {

        parent::__construct();

    }

    public function GetDatos(){

        $resultado=$this->bd_conect->query('select * from alumno');
        $datos=$resultado->fetch_all(MYSQLI_ASSOC);
        return $datos;

    }

    //la visualización que se realiza en index, debería ser un nuevo método Visualizar().
}

?>
```

```
<?php
const BD_H="localhost:3308"; //o la función define (BD_H,"localhost:3308");
const BD_U="root";
const BD_PW="";
const BD_NOM="ciclo";
const BD_C="UTF8";

?>
```

8. Actividad recogida en el Anexo I (MVC).
9. Utiliza el MVC para las actividades planteadas en RepasoSQL_modificado

10. Leer un fichero .html.

```
<?php

$nomfich="ficherohtml.html";
$fcont=fopen($nomfich,"r");
$contenido=fread($fcont,filesize($nomfich));
echo $contenido;
fclose($fcont);

-?>
```

```
<html>
<head>
<title> Probando...</title>
</head>
<body>
<h1>HOLA</h1>
<h1>¿Que tal?</h1>
</body>
</html>
```

HOLA
¿Que tal?

11. Antes de la inserción de datos en una BD capturados desde un formulario

En la UT 4 vimos cómo a través de un formulario es posible incluir código malicioso. Para evitarlo, desde PHP existen determinadas funciones que contribuyen a “satinizar” las entradas de información antes de que dicha información sea incorporada a una consulta en una BD. Una de ellas es **htmlspecialchars()**, convierte los caracteres especiales en entidades HTML; con ello se consigue que caracteres especiales como <,>,&,...sean sustituidos por <, >, &,...y con ello evitar que puedan ser interpretados como script maliciosos escritos en JavaScript por ej.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title> Comprobando vulnerabilidades...</title>
</Head>
<body>
<h3> </h3>
<br>
<!--FORMULARIO -->
<form action="seguir.php" method="post">
  ¿De qué color es el caballo blanco de Santiago?:
  <input type="text" name="dato">
  <span><font color="red"> * campo obligatorio.</font></span><br><br>
  <input type="submit" name="enviar">
</form>
</body>
</html>
```

```
<?php

//TRATAMIENTO

if (!empty($_POST['dato']))

    if ($_POST['dato']=="blanco")
        echo "!!!Acertaste!!!";
    else
        echo $_POST['dato'];

?>
```

```
<?php

//TRATAMIENTO

if (!empty($_POST['dato']))

    if ($_POST['dato']=="blanco")
        echo "!!!Acertaste!!!";
    else
        echo htmlspecialchars($_POST['dato']);

?>
```

Al ejecutar este código....

¿De qué color es el caballo blanco de Santiago?: * campo obligatorio.

Enviar

localhost:8080 dice
hackeado

Aceptar

<script>alert("hackeado")</script>

Si en lugar de realizar el tratamiento en otra página (seguir.php) se hiciera en la misma, es conveniente también que el valor del atributo action del formulario sea inicializado con el valor "<?php echo htmlspecialchars(\$_SERVER['PHP_SELF']);?>"

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title> Comprobando vulnerabilidades...</title>
</Head>
<body>
<h3> </h3>
<br>
<?php

//TRATAMIENTO

if (!empty($_POST['dato']))

    if ($_POST['dato']=="blanco")
        echo "!!!Acertaste!!!";
    else
        echo htmlspecialchars($_POST['dato']);

?>

<!--FORMULARIO -->
<form action="<?php echo htmlspecialchars($_SERVER['PHP_SELF']);?>" method="post">
    ¿De qué color es el caballo blanco de Santiago?:
        <input type="text" name="dato">
        <span><font color="red"> * campo obligatorio.</font></span><br><br>
        <input type="submit" name="enviar">
</form>
</body>
</html>
```