Práctica completa aplicación Web (I)

F

A lo largo del libro hemos visto muchos aspectos teóricos con algunos ejemplos prácticos, siempre sencillos y focalizados en el problema que estábamos tratando. En este apéndice desarrollaremos una pequeña aplicación PHP 5 o 6 pero completa y totalmente funcional.

Aplicación: Licencia de activación

El objetivo de esta aplicación es suministrar el código de activación de un programa de software adquirido en una tienda de informática o en una librería. El software se distribuye en un CD-ROM y se vende junto con un manual; obviamente, los desarrolladores quieren que sólo los compradores del libro tengan derecho al uso del software y, además, en un único equipo.

Solución propuesta

La solución requiere el uso de tres códigos:

- Un número de serie
- Un código identificador del equipo
- Una clave de activación

El número de serie se distribuye dentro del sobre del CD que viene pegado en la contra cubierta del libro. Es un número no consecutivo de 8 dígitos generado con un algoritmo matemático que genera una serie difícil de descubrir. El comprador deberá informar ese número en la primera ejecución del programa, que validará que sea un número correcto.

El código identificador del equipo lo determina el programa cada vez que se inicia la ejecución del software. Algo que identifica sin dudas a un equipo es su número MAC (el número de la tarjeta de red).

La clave de activación es un valor personalizado que se calcula a partir de los dos valores anteriores.

Tal como hemos dicho, cuando el comprador ejecuta por primera vez el software deberá suministrar el número de serie y, por su parte, el programa obtendrá automáticamente el valor MAC; esos dos datos quedan guardados en un registro interno a la espera que se suministre la clave de activación. Sin clave de activación el software no pasa de la primera pantalla de presentación.

Para obtener la clave de activación el comprador debe conectarse a la web de los fabricantes del software y dar sus datos personales junto con el número de serie y el código identificador del equipo. La página php que desarrollaremos interviene en este paso capturando toda esta información. Luego de un proceso de validación y registro en una base de datos de licencias, le enviará la clave de activación por correo electrónico.

Al recibir por correo la clave de activación el comprador puede suministrarla en la página de inicio del software, que la validará con el número de serie, que tiene almacenado en un registro, y con el código identificador del equipo, que se obtiene con una función que detecta el número MAC de la tarjeta de red. Si todo está bien almacenará el código de activación junto con el número de serie en un registro interno.

En cada inicio del programa se repite esta validación (sin necesidad que el usuario teclee la clave de activación, ya que ésta está en un registro). Si por alguna razón se cambió la tarjeta de red o porque la instalación se hizo en un segundo equipo, el proceso de instalación detectará que las claves no concuerdan.

Recordemos que la clave de activación se genera teniendo en cuenta el valor del número de serie y el número MAC, cualquier cambio hará que la clave de activación resulte inválida.

Nota: Si el comprador elimina la tarjeta de red del equipo, cosa habitual al usar equipos portátiles, provocará que la clave de activación no funcione y el software quedará inoperativo. Se podría mejorar el cálculo del código identificatorio del equipo utilizando otros elementos del equipo: datos del procesador, nombre del equipo, etc.

Organización física del proyecto

Nuestro proyecto podría estar todo debajo del directorio raíz del servidor pero eso generaría un caos en corto tiempo. Nuestro sitio web seguramente atenderá diferentes aplicaciones, por lo que lo mejor es crear debajo del directorio raíz un directorio para la aplicación, con un nombre que podría ser Licencia.

Debajo del directorio Licencia podemos crear una estructura organizativa que nos facilite el mantenimiento del sitio. En el sitio tendremos:

- Archivos php de uso normal; por uso normal se entiende que son páginas de uso sin restricciones (carpeta Licencia)
- Archivos php de definición de clases (carpeta Licencia\Objetos)

El código de esta aplicación se encuentra en la carpeta Parte 4/code/ Licencia.

Datos del proyecto

La base de datos del proyecto se denomina LicenciaDB y contendrá la siguiente tabla:

- Licencia

La tabla Licencia contendrá los datos personales del comprador y el juego de las tres claves (el número de serie, el código que identifica el equipo y la clave de activación asignada).

Cada vez que se inserta una nueva licencia se verifica que el número de serie ya no haya sido asignado a otro comprador.

Clases del proyecto

Utilizaremos dos clases: una para comunicarnos con la base de datos (AuxDB) y otra para representar el objeto Licencia.

Veamos ahora detalladamente cada una de las clases y analicemos qué métodos y propiedades deben gestionar.

Clase AuxDB

Aunque sea una clase auxiliar será necesario que la definamos en primer término, ya que la emplearemos en el resto de las clases para establecer la comunicación con el servidor MySQL. Esta clase estará codificada para entenderse con el gestor de base de datos MySQL. Si, por alguna razón, en algún momento posterior, quisiéramos cambiar nuestro sistema de base de datos no tendríamos que modificar el proyecto ni ninguna de sus clases funcionales (las que no son auxiliares): simplemente tendríamos que escribir el código de una nueva clase AuxDB para que se sepa comunicar, por ejemplo, con SQLite, SQL Server u Oracle.

Esa nueva clase AuxDB tendría que poseer los mismos métodos que la clase que reemplaza, de esa manera nos aseguramos que los objetos que la utilicen no intentarán ejecutar ningún método que no esté implementado. Para simplificar esta tarea se suele definir una interfaz que establece una especie de contrato vacío (sólo la lista de métodos, sin proveer su implementación) que cada clase que la utilice deberá cumplir. Por lo tanto, si definimos la interfaz y luego al crear cada clase establecemos que utiliza dicha interfaz nos aseguraremos que la clase implementa todos los métodos necesarios.

Por todo lo dicho, la definición de nuestra clase auxiliar no comienza con la definición de la propia clase sino con la definición de la interfaz.

Ya hemos dicho para qué nos servirá la clase auxiliar:

- Conexión con el gestor de base de datos (método conectar)
- Desconexión del gestor de base de datos (método desconectar)
- Ejecución de sentencias SQL (método ejecutarSQL), pudiendo ser sentencias que devuelven filas o no
- Gestión del conjunto de resultados de una consulta
    - Obtener la fila siguiente de una consulta (método obtenerFila)
    - Conocer la cantidad de filas obtenidas en una consulta (método cantidadFilas)

- Liberar el conjunto de resultados (método liberarRecursos)

Nota: el uso de interfaces es una inversión a futuro. En este pequeño proyecto de ejemplo podríamos omitir su uso y definir directamente la clase AuxDB sin hacer referencia a la interfaz; pero al hacerlo de esta manera nos resultará más fácil crear otras clases auxiliares que cumplan los mismos objetivos pero en otros entornos.

Nuestra interfaz guardada podría denominarse IAuxDB.php y estará ubicado dentro del directorio Objetos.

```
<?php
// IAuxDB.php

interface IAuxDB
{
// funciones que se deben implementar
// en la clase que elija utilizar esta interfaz

    function conectar();
    function desconectar();
    function ejecutarSQL($strsql);
    function siguienteFila($rst);
    function liberarRecursos($rst);
}

?>
```

La definición de la clase AuxDB la guardaremos en un archivo que denominaremos AuxDB.php y que almacenaremos dentro del directorio Licencia\Objetos. En este código hemos utilizado la extensión mysqli.

```
<?php
// AuxDB.php


class AuxDB implements IAuxDB
{
// esta clase es específica para trabajar con MySQL
// variable privada para guardar la cadena de conexión
private $strcon;


function conectar() {
    $this->strcon = mysqli_connect("localhost","root","0021",
          "LicenciaDB") or
          die("Error de aplicación: No conectó con la base de
```

```
                datos");
}

function desconectar() {
    mysqli_close($this->strcon);
}

function ejecutarSQL($strSQL) {
     print $strSQL . "<BR>";
    $result = mysqli_query($this->strcon, $strSQL);
    // Muestra el detalle del mensaje de error MySQL
    // esto no se debería dejar en una aplicación en producción
    if (!$result) {
         $msg  = 'Consulta inválida: ' . mysql_error() . "\n";
         $msg .= 'SQL: ' . $strSQL;
         die($msg);
    }

    return $result;
}

function siguienteFila($rst) {
    return mysqli_fetch_assoc($rst);
}

function cantidadFilas($rst) {
    return mysqli_num_rows($rst);
}

function liberarRecursos($rst) {
    mysql_free_result($rst);
}
}
?>
```

Clase Licencia

La clase Licencia nos permitirá crear objetos Licencia y, dada la definición del proyecto Licencia, será necesario que la clase Licencia tenga métodos que nos facilite la lectura e inserción de Licencias.

En el capítulo dedicado al estudio de clases y objetos ya hemos visto cómo definir una clase en PHP:

```
<?php
class Licencia
{
```

A continuación se definen las variables miembro o propiedades de la clase.

Propiedades de la clase Licencia

Las propiedades de la clase Licencia se corresponden con los campos de la tabla Licencia:

- **AClave1**: número de serie que nos permite definir de modo único a cada licencia de la tabla. Elegiremos un campo INT, no nulo. Será la clave primaria de la tabla.
- **AClave2**: es el código identificador del equipo del cliente. Es un campo VARCHAR de 64 caracteres, no puede tener valor nulo.
- **AClaveActiva**: es la clave de activación asignada. Es un campo VARCHAR de 4 caracteres y no puede tener valor nulo.
- **AApellidos**: es el campo para almacenar los apellidos o el nombre de la empresa del cliente. Es un campo VARCHAR de 50 caracteres y no puede tener valor nulo.
- **ANombre**: es el campo para almacenar el nombre del cliente. Es un campo VARCHAR de 50 caracteres y que podría tener valor nulo.
- **ADomicilio**: es un campo para informar el domicilio del comprador. No es un campo obligatorio, por lo que será un campo VARCHAR de 50 caracteres y que podría tener valor nulo.
- **ATeléfono**: es un campo para informar el teléfono del comprador. No es un campo obligatorio, por lo que será un campo VARCHAR de 50 caracteres y que podría tener valor nulo.
- **AEmail**: es un campo para informar el correo electrónico del comprador. Es un campo obligatorio, ya que el programa debe enviar la clave de activación por correo, por lo que será un campo VARCHAR de 50 caracteres y que no podrá tener valor nulo.
- **AObserv**: es un campo opcional para informar algún dato que quiera aportar el comprador. No es un campo obligatorio, por lo que será un campo TEXT y que podría tener valor nulo.
- **AFecha**: es un campo Date para almacenar la fecha de creación de la licencia.

Siguiendo con la definición de la clase en nuestro código php:

```
{

private $AApellidos;
private $ANombre;
```

```
private $ADomicilio;
private $ATeléfono;
private $AEmail;
private $AClave1;
private $AClave2;
private $ATexto;
private $AClaveActiva;

function getAClaveActiva() {return $this->AClaveActiva;}

function Licencia($p1, $p2, $p3, $p4, $p5, $p6, $p7, $p8)
{
    $this->AApellidos     = $p1;
    $this->ANombre        = $p2;
    $this->ADomicilio     = $p3;
    $this->ATeléfono      = $p4;
    $this->AEmail         = $p5;
    $this->AClave1        = intval($p6);
    $this->AClave2        = intval($p7);
    $this->ATexto         = $p8;

    // calcula la clave de activación con un algoritmo
    // que combina el valor de las dos claves
    $b =  intval((log($this->AClave1) + sin($this->AClave2))
          * 137549);

    $this->AClaveActiva = substr($b,2,4);
}
```

Ahora podemos crear la tabla siguiendo esta información. Podemos utilizar PHPMyAdmin con este código SQL:

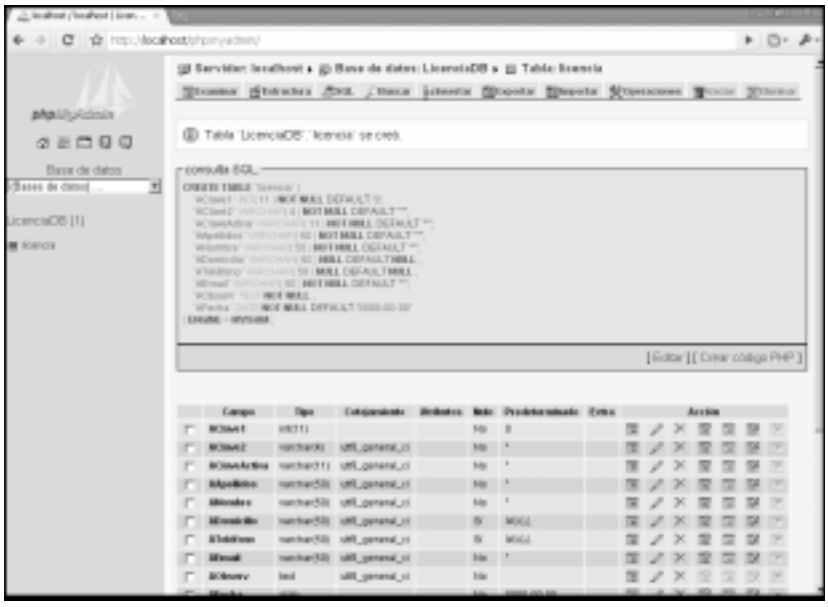
```
DROP TABLE IF EXISTS     'licencia';
CREATE TABLE        'licencia' (
  'AClave1'         int(11)          NOT NULL default '0',
  'AClave2'         varchar(4)       NOT NULL default '',
  'AClaveActiva'    varchar(4)       NOT NULL default '',
  'AApellidos'      varchar(50)      NOT NULL default '',
  'ANombre'         varchar(50)      NOT NULL default '',
  'ADomicilio'      varchar(50)      default NULL,
  'ATelefono'       varchar(50)      default NULL,
  'AEmail'          varchar(50)      NOT NULL default '',
  'AObserv'         text,
  'AFecha'          date             NOT NULL default '0000-00-00',
  PRIMARY KEY  ('AClave1')
) TYPE=MyISAM;
```

En la pantalla de creación de la tabla se nos pide que indiquemos la cantidad

total de campos (columnas) que tendrá la tabla (en este caso, 10).

Utilizamos el formulario que muestra en la siguiente figura para definir los campos según el criterio establecido y pulsamos el botón Grabar.

En la siguiente pantalla aparece la sentencia SQL que utilizó el asistente para crear la tabla (Create Table) y también se muestra una rejilla que nos permite realizar distintas acciones con los campos (Cambiar, Eliminar, Definir como clave primaria, Crear un índice, Indicar que no se admiten duplicados, etc.



Aprovechamos esta pantalla para definir AClave1 como clave primaria de la tabla Licencia.

Nota: entre los archivos de la aplicación (Parte 4/code/Licencia) se incluye el archivo creartable.sql, que tiene las sentencias SQL para crear la tabla sin necesidad de tener que crearlas manualmente.

Métodos de la clase Licencia

La clase Licencia nos permitirá crear objetos Licencia y, dada la definición del proyecto Licencia, será necesario que la clase Licencia tenga métodos que nos facilite la lectura e inserción de las licencias. Por lo que desarrollaremos los siguientes métodos:

- Leer
- Insertar

El método leer() nos permite verificar que la licencia no ha sido utilizada por otro cliente:

```
function leer()
{

    // Crear un objeto AuxDB
    $db = new AuxDB();
    $db->conectar();

    // sentencia Select para comprobar si existe el registro
    $sql = "Select * FROM Licencia Where AClave1 = " .
            $this->AClave1 . " ";

    // ejecución de la sentencia SQL
    $result = $db->ejecutarSQL($sql);
    $result = $db->cantidadFilas($result);

    // desconexión con el servidor de base de datos
    $db->desconectar();
    return $result;
}
```

La función insertar nos permite añadir nuevas licencias a la tabla Licencia. Este método presupone que las propiedades del objeto ya recibieron las asignaciones y sólo le queda insertar el registro en la tabla Licencia. Obsérvese el uso de la función mysql_escape_string que incluye los escapes necesarios en la cadena para que ésta sea apta para ser tratada por la función mysql_query(), por ejemplo, insertando los caracteres de escape en símbolos problemáticos como el apóstrofo. También se emplea la función Now() para actualizar el campo fecha.

```
function insertar()
{

    // Crear un objeto AuxDB
    $db = new AuxDB();
    $db->conectar();

    // sentencia INSERT para insertar una Licencia
    $sql = "Insert Into Licencia (AApellidos, ANombre, ADomicilio,";
    $sql.= " ATeléfono, AEmail, AClave1, AClave2, AObserv, ";
    $sql.= " AClaveActiva, AFecha) Values ('";
    $sql.= mysql_escape_string($this->AApellidos) . "', '";
    $sql.= mysql_escape_string($this->ANombre) . "', '";
    $sql.= mysql_escape_string($this->ADomicilio) . "', '";
    $sql.= mysql_escape_string($this->ATeléfono) . "', '";
    $sql.= mysql_escape_string($this->AEmail) . "', ";
    $sql.= mysql_escape_string($this->AClave1) . ", '";
    $sql.= mysql_escape_string($this->AClave2) . "', '";
    $sql.= mysql_escape_string($this->ATexto) . "', '";
    $sql.= mysql_escape_string($this->AClaveActiva) . "', ";
    $sql.=  "NOW() ) ";

    // ejecución de la sentencia SQL
    $db->ejecutarSQL($sql);

    // desconexión con el servidor de base de datos
    $db->desconectar();
}

}
```

Y éste es todo el código necesario para la clase.

Estructura de las páginas PHP

Ya hemos visto en la organización física del proyecto que tendremos dos carpetas en donde distribuiremos los archivos de la aplicación. La carpeta `Objetos` nos permite agrupar todas las clases del proyecto y en la carpeta raíz estarán las páginas php y los archivos de inclusión.

En la carpeta raíz del proyecto (Licencia) crearemos estos archivos:

- **header.php**: archivo de inclusión que contendrá la definición de las funciones comunes y el encabezamiento HTML de la página.
- **estilos.php**: archivo de inclusión que define los estilos utilizados en las páginas.

- **inicio.php**: contendrá la página de inicio del sitio. Es la que suministra el formulario de introducción de datos del cliente y del software.
- **footer.php**: archivo de inclusión que contendrá el cierre de las páginas HTML.
- **actualizar.php**: es la página que permite actualizar la base de datos.

header.php

En este archivo de inclusión definiremos la función interceptora __autoload() la que se llama automáticamente cuando se intenta crear un ejemplar de una clase que aún no se declaró. Ya la hemos visto en el capítulo dedicado a Clases y aquí se utilizará cada vez que se haga referencia por primera vez a una de las clases del proyecto.

```
<?php
// header.php

function __autoload($objeto)
{
    include("Objetos/".$objeto.".php");
}
?>
<html>
<head>
<title>Aplicación de ejemplo: Licencia de activación (capítulo 19)
</title>

<?php
include("estilos.php")
?>

</head>
<body>
<div align="center">
<table cellpadding="0" cellspacing="0" width="90%">
<tr><td colspan="3" class="título">Licencia de activación de
     software</td></tr>
<tr><td colspan="3"><br/></td></tr>

<tr>
  <td class="marco" width="80%">
```

estilos.php

En este archivo de inclusión codificaremos los estilos que utilizaremos en las páginas. Definiremos tres estilos:

```
<style type="text/css">
```

```
<!--
.marco { border-style: solid;
            font-family: Arial;
            font-size: 14px;
            border-color: blue;
            border-width: 3px;
            vertical-align: top;
            padding:3px;}
.título {font-family: Arial;
              font-size: 18px;
              text-align:center }
.datos { border : thin solid Black;
              font-family : Times;
              font-size : 18px;
              background-color : transparent;
              color: #000000;
}
-->
</style>
```

footer.php

En este archivo de inclusión simplemente cerraremos todas las etiquetas HTML que se han abierto en los archivos previos.

```
  </td>
</tr>
</table>
</div>
</body>
</html>
```

inicio.php

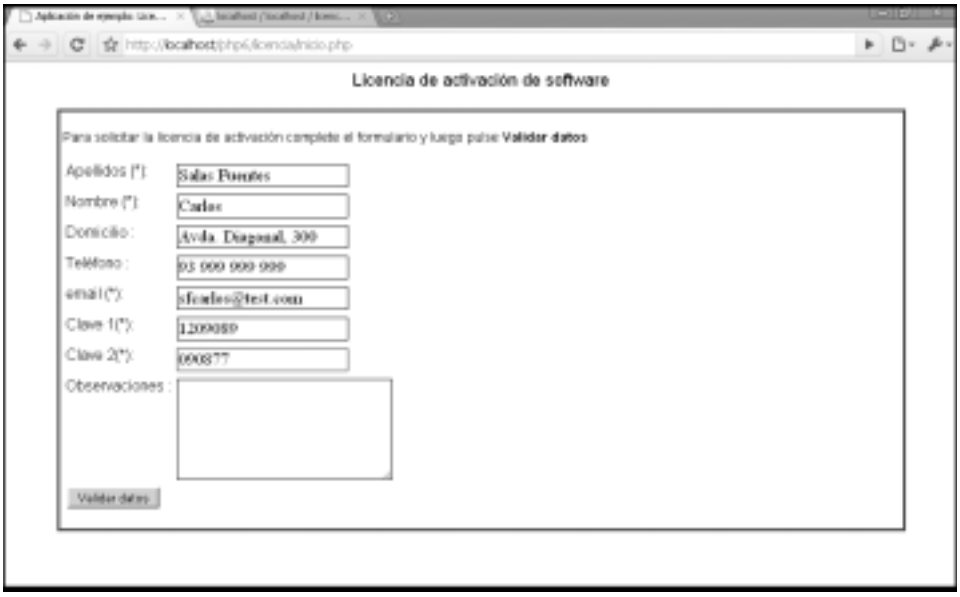
Este archivo php representa el inicio de la aplicación y el formulario principal de entrada de datos.

```
<?php
// inicio.php

include("header.php");

$oper = "";
if( isset( $_GET['oper'] ) ) {
    $oper = $_GET['oper'];
}


switch( $oper ){

    case "validar":
```

```
      // se eliminan las barras invertidas
      $apellidos    = stripslashes( $_POST['apellidos'] );
      $nombre       = stripslashes( $_POST['nombre'] );
      $domicilio    = stripslashes( $_POST['domicilio'] );
      $teléfono     = stripslashes( $_POST['teléfono'] );
      $email        = stripslashes( $_POST['email'] );
      $clave1       = stripslashes( $_POST['clave1'] );
      $clave2       = stripslashes( $_POST['clave2'] );
      $texto        = stripslashes( $_POST['texto'] );

 // valida que los campos obligatorios no estén vacíos

if( isset( $_POST['apellidos'] ) && !empty( $_POST['apellidos'] ) &&
    isset( $_POST['nombre'] ) && !empty( $_POST['nombre'] ) &&
    isset( $_POST['email'] ) && !empty( $_POST['email'] ) &&
    isset( $_POST['clave1'] ) && !empty( $_POST['clave1'] ) &&
    isset( $_POST['clave2'] ) && !empty( $_POST['clave2'] ) ) {

    print("<br>Para solicitar la licencia de activación pulse el
           botón <b>Confirma datos</b>");
    print("<br>Para corregir los datos pulse el botón Atrás del
           navegador <br><br>");
    $fase = "registrar";
    $pagref = "actualizar";
    $submite = "Confirma datos";

}
else {
    // si hay errores se muestra nuevamente la misma página
    print("<br><b>Faltan informar datos obligatorios (*)</
          b><br><br>");
    $fase = "validar";
    $pagref = "inicio";
    $submite = "Validar datos";

}

break;


default:
    // si entra por primera vez muestra el formulario vacío
    print("<br>Para solicitar la licencia de activación complete
         el formulario y luego pulse <b>Validar datos<br/><br/>");
    $apellidos = "";
    $nombre ="";
    $teléfono = "";
    $clave1 ="";
   $domicilio = "";
   $clave2 = "";
   $email = "";
```

```
    $texto = "";
    $fase = "validar";
    $pagref = "inicio";
    $submite = "Validar datos";

}

// se crea el formulario

print("<form action=\"$pagref.php?oper=$fase\" method=\"post\">");
print("<table>");
print("<tr><td valign=\"top\">Apellidos (*):</td><td><input
     type=\"text\" name=\"apellidos\" class=\"datos\" tabindex=\"1\"
     size\"5\" value = \"$apellidos\"></td></tr>\n");
print("<tr><td valign=\"top\">Nombre (*):</td><td><input
     type=\"text\" name=\"nombre\"   class=\"datos\" tabindex=\"2\"
     size\"5\" value = \"$nombre\"></td></tr>\n");

print("<tr><td valign=\"top\">Domicilio :</td><td><input
     type=\"text\" name=\"domicilio\"  class=\"datos\"
     tabindex=\"3\" size\"5\" value = \"$domicilio\"></td></tr>\n");

print("<tr><td valign=\"top\">Teléfono :</td><td><input type=\"text\"
     name=\"teléfono\" class=\"datos\" tabindex=\"4\" size\"5\"
     value = \"$teléfono\"></td></tr>\n");

print("<tr><td valign=\"top\">email (*):</td><td><input type=\"text\"
     name=\"email\" class=\"datos\" tabindex=\"5\" size\"5\" value =
     \"$email\" ></td></tr>\n");

print("<tr><td valign=\"top\">Clave 1(*):</td><td><input
     type=\"text\" name=\"clave1\" class=\"datos\" tabindex=\"6\"
     size\"8\" value = \"$clave1\" ></td></tr>\n");

print("<tr><td valign=\"top\">Clave 2(*):</td><td><input
     type=\"text\" name=\"clave2\" class=\"datos\" tabindex=\"7\"
     size\"4\" value = \"$clave2\"></td>\n");

print("<tr><td valign=\"top\">Observaciones :</td><td colspan =\"2\">
     <textarea name=\"texto\" rows=\"5\" cols=\"30\" class=\"datos\"
     tabindex=\"8\" >$texto</textarea></td></tr>\n");

print("<td><input type=\"submit\" value=\"$submite\"
tabindex=\"9\"></td></tr>");
print("</table>");
print("</form>");

include("footer.php");
?>
```

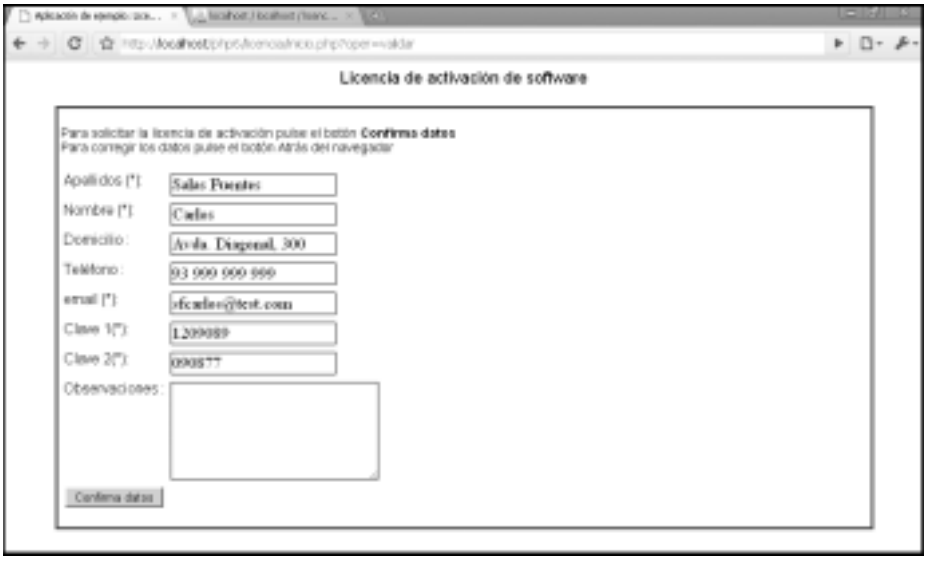
Como ya tenemos codificados todos los archivos de inclusión y clases que intervienen en esta página podemos comprobar su funcionamiento utilizando nuestro navegador y accediendo a http://localhost/PHP6/Licencia/inicio.php. Tendríamos que visualizar la siguiente pantalla:



Al pulsar el botón Validar datos se realiza envía el formulario a la página referenciada (que es la propia página inicio.php). En esta segunda ejecución de la pagina se realiza una validación formal del formulario, básicamente, que los campos obligatorios estén informados, que la dirección de correo electrónico tenga un formato correcto y que el número de serie sea válido.

Si no se detectan errores se da la posibilidad de confirmar los datos y solicitar la clave de activación. Esto lo hará la página actualizar.php.

Pero podrían detectarse errores, en tal caso se muestra nuevamente el formulario con un mensaje que indica que existen errores en la información. El cliente puede modificar los campos y volver a pulsar el botón Validar datos.



actualizar.php

Este archivo php recibe los datos ya comprobados del formulario y realiza una segunda verificación. Esta vez accede a la base de datos para verificar que la licencia ya no haya sido utilizada previamente, en cuyo caso genera un mensaje de error.

Si no detecta errores, realiza el cálculo de la clave de activación y registra toda la información en la base de datos.

Se podría mostrar la clave de activación por pantalla, pero como medida de seguridad se envía la clave por correo electrónico a la dirección informada por el cliente.

```
<?php
// actualizar.php

include("header.php");

// se eliminan las barras invertidas
$_POST['apellidos'] = stripslashes( $_POST['apellidos'] );
$_POST['nombre']    = stripslashes( $_POST['nombre'] );
$_POST['domicilio'] = stripslashes( $_POST['domicilio'] );
$_POST['telefono']  = stripslashes( $_POST['telefono'] );
$_POST['email']     = stripslashes( $_POST['email'] );
$_POST['clave1']    = stripslashes( $_POST['clave1'] );
$_POST['clave']     = stripslashes( $_POST['clave'] );
$_POST['texto']     = stripslashes( $_POST['texto'] );

// crea un objeto Licencia
$lic = new Licencia( $_POST['apellidos'],
```

```
                              $_POST['nombre'],
                              $_POST['domicilio'],
                              $_POST['teléfono'],
                              $_POST['email'],
                              $_POST['clave1'],
                              $_POST['clave2'],
                              $_POST['texto'] );

 if ( $lic->getAClaveActiva() <> "")  {
    if ($lic->Leer() <> 0)     {
         print "Esa clave ya fue asignada. Contacte con el vendedor
                del software.";
         exit;
    }
 }

 if ( $lic->getAClaveActiva() <> "")  {

    $lic->insertar();
    $claveactiva = $lic->getAClaveActiva();

    print("Se añadió su pedido de licencia. <br>");

    $to = $_POST['email'];
    $subject = "Licencia de activación de InfoSoft";
    $body = "<html><head><title>Clave de activación de INFOSOFT. ";
    $body.= ":</title></head>";
    $body .= "<body>Su clave de activación es $claveactiva ";
    $body.= "<br><br>Cualquier problema envíe un email ";
    $body .= "soporte@infosoft.com</body></html>";
    $header = "MIME-Version: 1.0\n";
    $header .= "Content-type: text/html; charset=iso-8859-1\n";
    $header .= "From: Edgar DAndrea <soporte@infosoft.com>\r\n";
    mail($to, $subject, $body, $header);
   print ("Recibirá la clave de activación en su correo  $to");
 }
 else {
    print "<br> No se pudo generar clave de activación.<br> Posible
         problema con el número de serie suministrado. <br> ";
    print "Si piensa que es un error, Póngase en contacto por
         teléfono (999 999 999)";
 }

include("footer.php");
?>
```

Al realizarse un registro correcto se muestra la siguiente pantalla:

Pero la página actualizar también podría detectar errores, en tal caso no insertaría ningún registro en la tabla, no enviaría el mensaje de correo electrónico y simplemente informaría el error detectado, tal como se muestra en la siguiente figura: