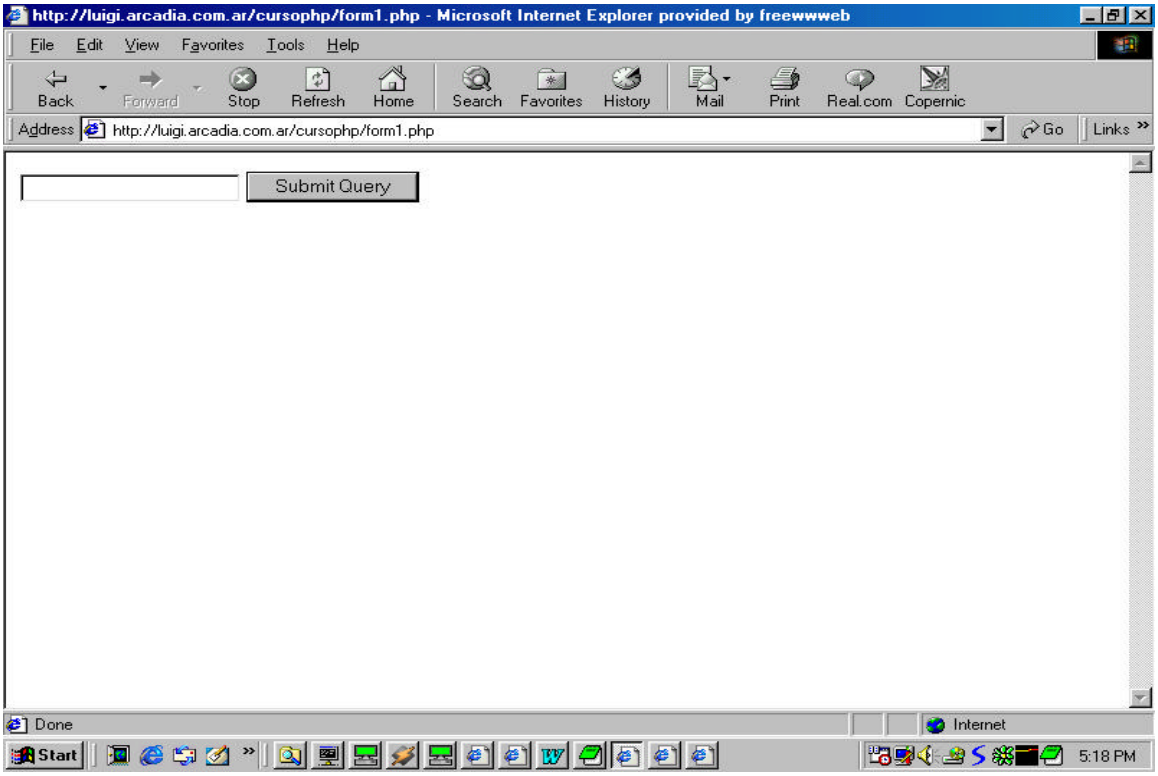
Capítulo 3: Manejo de Forms.

El mecanismo básico de interacción entre el usuario y un web-site esta dado por el uso de formularios html, el server envía un formulario que el browser muestra en pantalla permitiendo al usuario ingresar datos, luego los datos en el formulario viajan al server en el próximo request realizado por el browser para ser procesados en el mismo. La respuesta del server suele depender de los datos recibidos en el formulario.

El siguiente es un ejemplo de formulario en HTML usando un campo de entrada de tipo “text”

```
<FORM ACTION=”procesar.php”  METHOD=”POST”>
<INPUT TYPE=”text” NAME=”texto”>
<INPUT TYPE=”submit” NAME=”proc”>
</FORM>
```

Este formulario HTML se ve en pantalla de la siguiente manera:



Una vez que el usuario ingresa un texto y presiona el botón de submit el browser genera un request con método “Post” al script “procesar.php” que es el script que se va a encargar de procesar los datos ingresados en el formulario.

Dentro del script php los datos del formulario se reciben en variables php que tienen el mismo nombre que los indicados con “NAME” en el formulario, en este caso el script recibe $texto con el texto tipeado por el usuario en el formulario.

El script que recibe el formulario podría por ejemplo ser:
(procesar.php)

```
<?
print (“El valor ingresado en el formulario es: $texto <BR />”);
?>
```

En PHP es posible que un form se procese a si mismo, esto lo podemos hacer de la siguiente manera:
(form1.php)

```
<?
if(isset($proc)) {
  print(“el valor ingresado es: $texto”);
} else {
?>

<FORM ACTION=”form1.php”  METHOD=”POST”>
<INPUT TYPE=”text” NAME=”texto”>
<INPUT TYPE=”submit” NAME=”proc”>
</FORM>

<?
} //Esto cierra el else que abrimos arriba.
?>
```

Notar que el nombre del script que muestra el formulario es el mismo que el script usado en “action” para procesarlo, la instrucción isset de PHP devuelve true si la variable esta seteada, para un formulario si el usuario presiona el botón de submit se setea automáticamente la variable que corresponde al “NAME” del botón submit del formulario, por eso preguntamos si esta seteado $proc para saber si hay que mostrar el formulario o procesarlo. Podría también procesarse el formulario y a su vez mostrarlo o mostrar otro distinto, las variantes dependen de que es lo que se quiere hacer.

Text type:

Para ingresar texto mediante un formulario html se usa el tag input con atrobuto type=”text”, los atributos disponibles son:

Atributos:

maxlength=”numero”	Cantidad máxima de caracteres que se pueden ingresar
name=”text”	Nombre de la variable php que recibirá el valor
size=”numero”	Tamaño del campo de entrada a mostrar en pantalla
value=”texto”	Valor inicial a mostrar en el campo de entrada (default)

Hidden Type:

El tag input con type=”hidden” funciona en forma idéntica a un tipo “text” con la salvedad de que no se muestra en pantalla, esto es útil para pasar variables entre formularios o guardar variables “ocultas” en un formulario.

Checkboxes:

Los checkboxes son campos de entrada que soportan solamente los estados de seteado o no. Para ello se usa el tag input con type=”checkbox” , los atributos disponibles son:

Atributos:

Checked	Si el atributo esta presente el checkbox aparecerá marcado por default.
name=”text”	Nombre de la variable php que recibe el valor.
value=”text”	Valor que toma la variable si esta seteada, el default es “on”

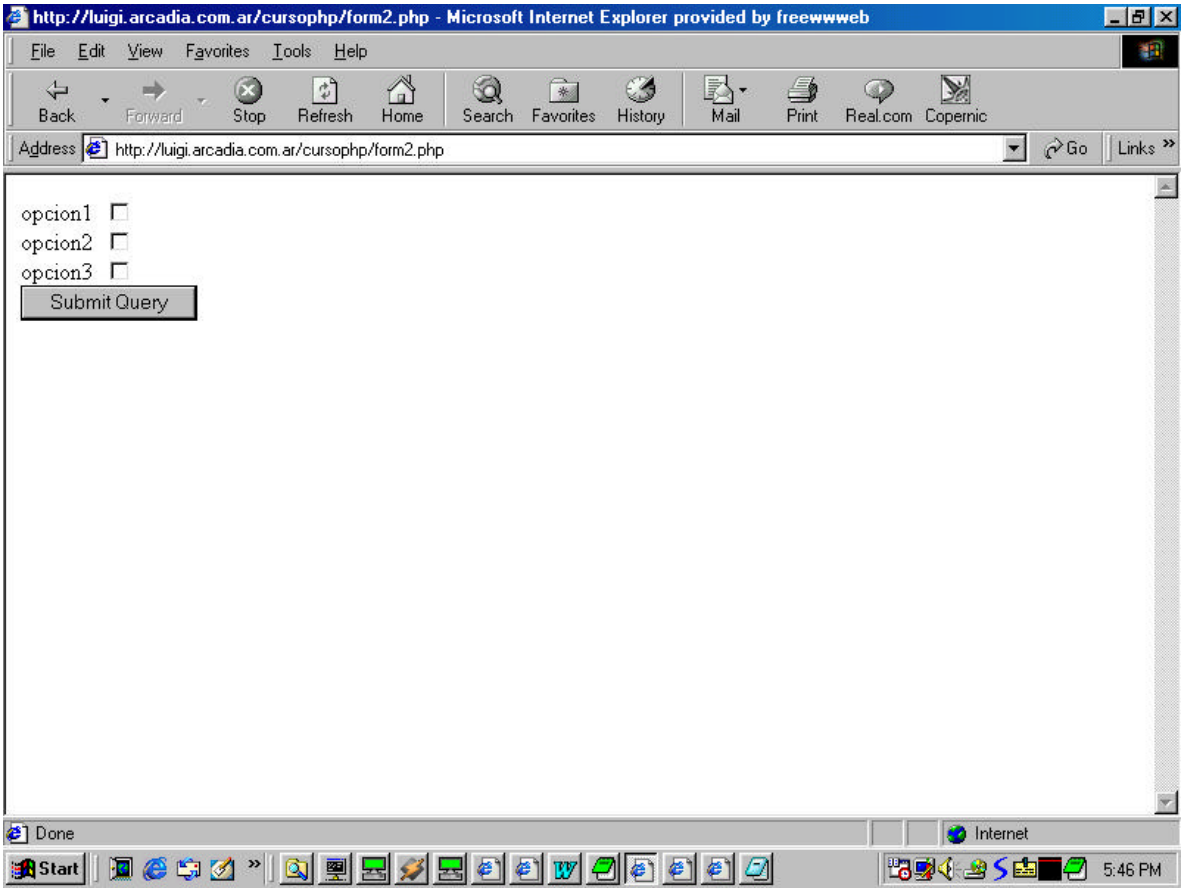
El script que recibe los resultados sólo recibe los nombres de los checkboxes que están seteados, es común en php generar una lista de checkboxes a partir de un vector, veamos un ejemplo:

(form2.php)

```
<FORM ACTION=”form2.php” METHOD=”post”>
<?
$vector=array(“opcion1”,”opcion2”,”opcion3”);
for ($i=0;$i<count($vector);$i++) {
  print(“$vector[$i]”);
  ?>
 <input type=”hidden” name=”valor[<?print($i);?>]”
value=”<?print(“$vector[$i]”);?>”>
  <input type=”checkbox” name=”vector[<?print($i);?>]”> <br>
  <?
}
?>
<INPUT TYPE=”submit” name=”proc”>
</FORM>
```

Este es un ejemplo muy útil en el cual el formulario html no es siempre el mismo sino que es generado dinámicamente desde php en base a por ejemplo el contenido de un vector.

El formulario se muestra en el browser de la siguiente manera:



Y el código html que recibe el browser para mostrar el formulario (que se genera en el servidor) es:

```
<FORM ACTION="form2.php" METHOD="post">
opcion1 <input type="hidden" name="valor[0]" value="opcion1">
<input type="checkbox" name="vector[0]"> <br>
opcion2 <input type="hidden" name="valor[1]" value="opcion2">
<input type="checkbox" name="vector[1]"> <br>
opcion3 <input type="hidden" name="valor[2]" value="opcion3">
<input type="checkbox" name="vector[2]"> <br>
<INPUT TYPE="submit" name="proc">
</FORM>
```

Observar el uso de campos de texto ocultos para indicar cual es el valor de un textbox en caso de estar seleccionado, también podríamos haber usado el campo value de los checkboxes, es otra forma de hacer lo mismo. Le podemos agregar al formulario la opción de mostrar cuales son los checkboxes seleccionados usando: (Agregar este código al principio de form2.php)

```
<?
if(isset($proc)) {
  for ($i=0;$i<count($valor);$i++) {
    if(isset($vector[$i])) {
      if($vector[$i]==”on”) {
          print(“$valor[$i] viene seleccionado”);
     }
    }
  }
}
?>
```

Como resultado el script informa cuales son los checkboxes que han sido seleccionados por el usuario y cuales no. El script completo es:

(form2.php)

```
<?
if(isset($proc)) {
  for ($i=0;$i<count($valor);$i++) {
    if(isset($vector[$i])) {
      if($vector[$i]=="on") {
          print("$valor[$i] viene seleccionado");
     }
    }
  }
}
?>

<FORM ACTION="form2.php" METHOD="post">
<?
$vector=array("opcion1","opcion2","opcion3");
for ($i=0;$i<count($vector);$i++) {
  print("$vector[$i]");
  ?>
 <input            type="hidden"            name="valor[<?print($i);?>]"
value="<?print("$vector[$i]")
;?>">
  <input type="checkbox" name="vector[<?print($i);?>]"> <br>
  <?
}
```

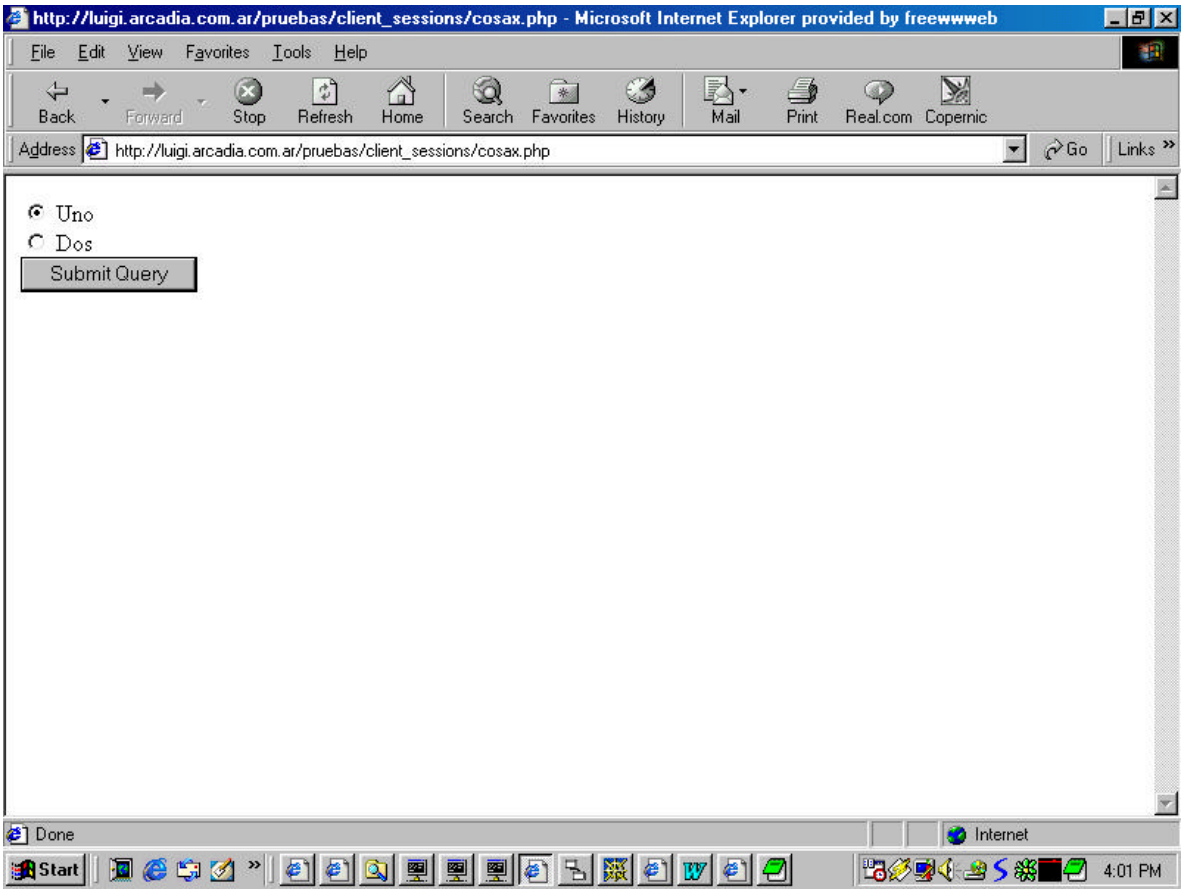
Radio Buttons:

Un radio button también es un tipo de botón de 2 estados (encendido-apagado) pero estos pueden agruparse de forma tal que sólo un radio-button del grupo pueda estar prendido. El nombre de radio button fue puesto por su funcionamiento parecido a los botones de las viejas radios de automóviles, que siempre debía estar presionado uno de los botones y nunca (mientras funcionaba correctamente) podían estar dos presionados al mismo tiempo.

Los atributos son checked, name y value. Todos los radio buttons del mismo nombre pertenecen al mismo grupo, es decir que de todo el grupo sólo uno podrá ser seleccionado. El value del botón determina cual es el radio elegido del grupo. Ejemplo:

```
<?
 if(isset($proc)) {
   print("El radio seteado es $grupo <BR />");
 }
?>

<FORM ACTION="form3.php" METHOD="post">
<input type="radio" name="grupo" value="uno" checked> Uno <br>
<input type="radio" name="grupo" value="dos"> Dos <br>
<INPUT TYPE="submit" name="proc">
</FORM>
```



El formulario que se despliega en pantalla es:

Y solamente uno de los dos radio buttons puede estar habilitado.

Image:

Es posible reemplazar el boton submit por una imagen gif o jpg de forma de darle al botón el look and feel que se quiera, esto se hace con el tag input type=”image” los atributos son:

name=”texto”	Cumple la misma función que el atributo name de submit.
src=”url”	URL de la imagen a mostrar, ejemplo: “images/boton.jpg”

TextArea:

Un textarea permite ingresar texto en una caja de formato mayor a un input type=”text” , la notación de un textarea es distinta a la de un tag de input. Los atributos son los siguientes:

cols=”numero”	Número de columnas del textarea
rows=”numero”	Número de líneas
name=”texto”	Nombre de la variable que recibe el texto
wrap=”off/virtual/physical”	Forma en la cual se cortan las palabras cuando termina cada línea, off elimina el corte de palabras, virtual muestra los cortes pero estos no son transmitidos al server, physical corta las palabras y además transmite los saltos de línea al server.

Ejemplo:

<textarea name=”texto” cols=”20” rows=”10”>
</textarea>

Entre el tag que abre y cierra si se desea se puede poner texto que se muestra en el textarea y el usuario puede modificar.

File Uploads

El último tipo de dato que se puede transferir al server usando un formulario es un archivo, este es un tipo de transferencia especial pues implica generar un archivo en el file-system del web-server a partir de un archivo que el usuario selecciona desde su disco local.

HTML soporta el upload de archivos usando el tag <input> con type=”file”, este tipo de input genera un boton “browse” en el browser que permite al usuario seleccionar un archivo desde su file-system local (usando una caja de navegación por los discos standard del sistema operativo).

El formulario para subir un archivo es de la forma:

```
<FORM ENCTYPE="multipart/form-data" ACTION="upload.php" METHOD=POST>
<INPUT TYPE="hidden" name="MAX_FILE_SIZE" value="1000">
Send this file: <INPUT NAME="userfile" TYPE="file">
<INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

Como puede verse hay un campo oculto que indica cual es el limite máximo de tamaño que se puede subir, este valor es chequeado en el “cliente”, además PHP dispone de una variable que se inicializa en el archivo de configuración de php (en gral /var/lib/php.ini) allí se limita el tamaño máximo de los uploads al llegar al “server”

El script upload.php que recibe los datos del formulario recibe las siguientes variables:

- $userfile – Path del archivo almacenado en el server.
- $userfile_name – Nombre del archivo segun el usuario
- $userfile_size – Tamaño del archivo subido
- $userfile_type – Mime type del archivo, por ejemplo image/gif

El script que recibe el archivo es responsable de hacer lo que corresponda con el mismo ya que en general el archivo se almacena en un directorio temporal y es eliminado una vez que termina el script. El script debe almacenar el archivo en una base de datos, moverlo a un directorio permanente, tomar datos de el o realizar el procesamiento que corresponda.