Cómo hacer selects dependientes con PHP, jQuery y AJAX

El mundo del desarrollo de las páginas web es un campo que está en continuo movimiento. Día tras día aparecen nuevas tecnologías orientadas a facilitar el desarrollo de los portales o bien para dotarlas de mayor funcionalidad. A pesar de esto, siempre hay ciertas funciones que se vienen utilizando desde hace mucho tiempo y que no pasan de moda. Un ejemplo son los llamados selects dependientes que serán los protagonistas a lo largo de nuestro White Paper de este mes.

¿Qué son los selects dependientes?

Puede ser que algunos de vosotros no hayáis escuchado hablar de los selects anidados, pero seguro que los habéis utilizado en muchas ocasiones al cumplimentar los datos de algún formulario en alguna página web.

Los selects dependientes consisten en el uso de dos o más selects cuyos valores se van cargando dependiendo de la opción elegida en el select anterior. Es muy utilizado para los desplegables de las provincias y localidades. Dependiendo de la provincia que se elija, las localidades que aparecerán en el segundo select serán diferentes. Pero éste no es el único uso, sino que se puede utilizar para otros muchos casos, por ejemplo para marcas de coches y modelos, o cursos y alumnos matriculados, por citar algunos.

A continuación veremos cómo podemos generar nuestros propios selects dependientes.

Bases de datos y tablas

Como hemos comentado anteriormente, es muy habitual utilizarlo para cargar en otro select las localidades de la provincia seleccionada, y será precisamente este el ejemplo sobre el que trabajaremos.

Lo primero que debemos hacer será crear las tablas en una base de datos que contendrá la información de las provincias y de las localidades. En este caso tendremos una tabla para las provincias y otra para las localidades. El código SQL para generar estas dos tablas es el siguiente:

Tabla provincias

```
CREATE TABLE IF NOT EXISTS `provincias` (
  `id_provincia` smallint(6) DEFAULT NULL,
  `provincia` varchar(30) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Tabla municipios

```
CREATE TABLE IF NOT EXISTS `municipios` (
`id_municipio` smallint(6) unsigned NOT NULL,
  `id_provincia` smallint(6) NOT NULL,
  `cod_municipio` int(11) NOT NULL,
  `DC` int(11) NOT NULL,
  `nombre` varchar(100) NOT NULL DEFAULT ''
) ENGINE=InnoDB AUTO_INCREMENT=8117 DEFAULT CHARSET=utf8;
```

Si nos fijamos en el código anterior, la tabla municipios tiene el campo '**id_provincia**' que hace referencia a la clave primaria de la tabla provincias.

Archivo dbConfig.php

Una vez que tengamos creada las tablas de la base de datos, nos centraremos en el archivo PHP que se encargará de realizar la conexión con la base de datos. Este código es parte fundamental, ya que sin él no podríamos ejecutar ninguna sentencia SQL para traernos la información.

El código que tendrá este archivo será el siguiente:

```
<?php
$dbHost = 'localhost';
$dbUsername = 'root';
$dbPassword = '';
$dbName = 'municipios';

//Conectamos y seleccionamos la base de datos
$db = new mysqli($dbHost, $dbUsername, $dbPassword, $dbName);

if ($db->connect_error) {
    die("Conexión fallida: " . $db->connect_error);
}
```

Las cuatro primeras líneas de este código las utilizamos para indicar los datos que usaremos para conectarnos a la base de datos.

A continuación, nos conectamos a la base de datos mediante la llamada al constructor "**mysqli**" al que le pasaremos como parámetros los datos de conexión a la base de datos. Las últimas líneas sirven para mostrar un mensaje de error al intentar conectarnos. Este error puede ser debido por utilizar datos erróneos o por un problema con el servidor de la base de datos.

Archivo index.php

En este archivo será donde pintemos los dos selects de nuestro ejemplo. El de provincias lo cargaremos al iniciar el programa, mientras que el de localidades, lo haremos mediante una llamada AJAX que se ejecutará cuando se cambiar el valor que aparezca en el campo provincias.

Lo primero que nos encontraremos será el código que se encargará de pintar el select con los datos de las provincias.

```
<?php
    include('dbConfig.php');
    $query = $db->query("SELECT * FROM provincias ORDER BY provincia ASC");
    $rowCount = $query->num_rows;
    ?>
    <select name="provincia" id="provincia">
        <option value="">Select Country</option>
```

```
        <?php
        if($rowCount > 0){
            while($row = $query->fetch_assoc()){
                echo '<option value="'.$row['id_provincia'].'">'.$row['provincia'].'</option>';
            }
        }else{
            echo '<option value="">Provincia no disponible</option>';
        }
        ?>
    </select>

    <select name="municipio" id="municipio">
        <option value="">Selecciona provincia primero</option>
    </select>
```

En este código lo primero que nos encontramos es la llamada al archivo de conexión a base de datos. Esto se hace mediante la instrucción:

```
include('dbConfig.php');.
```

A continuación definimos la instrucción SQL que se encargará de recuperar los nombres de las provincias almacenadas en bases de datos.

```
$query = $db->query("SELECT * FROM provincias ORDER BY provincia ASC");
```

También nos interesa saber el número de filas devueltas en la consulta a base de datos ya que si es 0, significa que no hay información y por tanto no pintamos el select de provincias. Este dato lo conseguimos mediante la ejecución de la siguiente línea:

```
$rowCount = $query->num_rows;
```

Si el número de filas devueltas es mayor que 0, entonces pintamos el select con todos sus datos. El código encargado de hacer esto es:

```
if($rowCount > 0){
                while($row = $query->fetch_assoc()){
                echo '<option
                value="'.$row['id_provincia'].'">'.$row['provincia'].'</option>';
        }
}
```

Las últimas líneas del código se encargarán de pintar el select de municipios vacío sin información:

```
<select name="municipio" id="municipio">
                <option value="">Selecciona provincia primero</option>
</select>
```

Archivo ajaxData.php

Este otro archivo será llamado por AJAX cada vez que se modifique el valor del select de las provincias. Se encargará de recuperar los datos de los municipios en base de datos.

```
<?php
include('dbConfig.php');

if(isset($_POST["id_provincia"]) && !empty($_POST["id_provincia"])){
    $query = $db->query("SELECT * FROM municipios WHERE id_provincia =
".$_POST['id_provincia']." ORDER BY nombre ASC");

    $rowCount = $query->num_rows;

    if($rowCount > 0){
        echo '<option value="">Select municipios</option>';
        while($row = $query->fetch_assoc()){
            echo '<option
value="'.$row['id_municipio'].'">'.$row['nombre'].'</option>';
        }
    }else{
        echo '<option value="">Municipio no disponible</option>';
    }
}
```

En este caso, el código es muy similar al visto en el punto anterior pero con dos pequeñas diferencias:

1. La consulta a base de datos varía ya que aquí se realiza para traerse la información de los municipios.
2. Se comprueba que en la llamada AJAX se haya incluido el parámetro "**id_provincia**", que es utilizado para traerse la información de los municipios de la provincia seleccionada.

Llamada AJAX en el archivo index.php

Por último, en el archivo **index.php** también hay que incluir la llamada AJAX que se ejecutará cada vez que se seleccione una provincia. Esta llamada la haremos utilizando jQuery:

```
<script src="jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
    $('#provincia').on('change',function(){
        var provinciaID = $(this).val();
        if(provinciaID){
            $.ajax({
                type:'POST',
                url:'ajaxData.php',
                data:'id_provincia='+provinciaID,
                success:function(html){
                    $('#municipio').html(html);
                }
            });
        }else{
```

```
            $('#municipio').html('<option value="">Selecciona una provincia primero</option>');
        }
    });

});
</script>
```

Lo primero que hacemos es incluir la librería jQuery, cosa que hacemos con la primera línea del código. A continuación, cuando el DOM de la página está cargado, detectaremos mediante la función "**on**" de jQuery que se ha modificado el valor del select de provincias. Cuando esto ocurre, obtenemos el id de la provincia mediante el código.

```
var provinciaID = $(this).val();
```

Si la variable "**provinciaID**" tiene un valor asignado, entonces se hace la llamada AJAX al archivo "**ajaxData.php**" al que se le pasa por POST la variable "**id_provincia**", a la que le asignamos el valor almacenado en "**pronvinciaID**".

```
type:'POST',
url:'ajaxData.php',
data:'id_provincia='+provinciaID,
```

Si todo ha ido bien, el select de municipios se cargará con los datos devueltos por el archivo "**ajaxData.php**". El código que se encarga de pintar la información es el siguiente.

```
success:function(html){
            $('#municipio').html(html);
}
```

Como hemos podido ver a lo largo de nuestro libro blando de este mes, crear selects anidados resulta muy sencillo. Ahora solo falta que lo probéis en vuestros proyectos.