Documentación Lab 5 - AJAX

Sistemas Web | Leroy Deniz

Solución Opcional 1.

Creé un archivo **CountXMLQuestions.php** que, pasándole el email de la URL como parámetro, carga el XML Questions.xml y lo recorre todo, contando todos los nodos ingresados y, además, pregunta si el atributo mail de cada uno se corresponde con el email ingresado. Finalmente hace un *echo* de los resultados de la forma en que quiero que se muestre cuando traiga ese archivo. Adjunto el código:

```
<?php

    # necesito el mail para poder saber, de todas las preguntas, cuáles son del usuario que consulta
    $email = $_GET['email'];

    if(!$xml = simplexml_load_file('../xml/Questions.xml')){
        echo "<script>alert('No se ha podido cargar el XML');</script>";
    } else {
        $counterTotal=0;
        $counterPropias=0;

        foreach ($xml as $pregunta) {
            $counterTotal = $counterTotal +1;
            if($pregunta['author']==$email){
                $counterPropias = $counterPropias +1;
            }
        }
        echo "Todas las preguntas: ".$counterTotal." | Mis preguntas: ".$counterPropias;
    }

?>
```

Hasta acá la información a mostrar, ahora vamos con la forma de mostrarla y actualizarla en la vista **HandlingQuizesAjax.php**. Aquí creé un div con id **qcounter** que deberá mostrar lo que imprima el php anterior. Para eso utilizo un archivo de JavaScript **CountQuestions.js** que se encargará, utilizando la función setInterval cada 2 segundos. Adjunto código:

```
setInterval(function() {

        /* creamos la variable con la solicitud */
        xhr = new XMLHttpRequest();

        /* indico qué traer y le paso el parámetro de email para saber las preguntas del usuario */
        xhr.open('GET','../php/CountXMLQuestions.php?email='+$("#email").val(), true);

        /* le digo qué hacer cuando llegue la respuesta */
        xhr.onreadystatechange = function(){
            if(xhr.readyState == 4 && xhr.status == 200) {
```

```
                $("#qcounter").html(xhr.responseText);
                //document.getElementById('qcounter').innerHTML = xhr.responseText;
            }
        }
        xhr.send('');

}, 2000);
```

Entonces, cada dos segundos se actualiza en la vista el número de preguntas en el XML.

Solución Opcional 2.

De la misma forma que en el Opcional 1, uso dos archivos PHP para consultar y mostrar la información de los usuarios conectados. Para esto, va a buscar a un archivo UserCounter.xml que tiene la siguiente estructura:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Users>
    <totalOfUsers>0</totalOfUsers>
</Users>
```

Como no me interesa saber quién está, sino cuántos, basta con tener un único nodo que lleve el seguimiento de la cantidad de usuarios al mismo tiempo. Esto va a estar delimitado además a que el usuario haga login y logout como banderas para incrementar o decrementar el contenido de ese único nodo.

Los dos archivos que mostrarán se encargarán de manipular el XML son los siguientes con su respectivo código. Cuando sean invocados, aumentarán a uno o reducirá a uno el nodo contador.

IncreaseGlobalCounter.php

```
<?php
    $email = $_GET['email'];
    if(!$xml = simplexml_load_file('../xml/UserCounter.xml')){
        echo "<script>alert('No se ha podido cargar el XML en IncreaseGlobalCounter.php');</script>";
    } else {
        # toma el primer y único elemento del xml que lleva el control del total de usuarios y le incrementa uno
            $suma = $xml->totalOfUsers;
            $suma = $suma + 1;
            $xml->totalOfUsers = $suma;
        # guardo el nuevo xml
        $xml->asXML('../xml/UserCounter.xml');
    }
    echo "<script>document.location.href='Layout.php?email=$email';</script>";
?>
```

DecreaseGlobalCounter.php

```
<?php
    if(!$xml = simplexml_load_file('../xml/UserCounter.xml')){
        echo "<script>alert('No se ha podido cargar el XML en DecreaseGlobalCounter.php');</script>";
    } else {
        # toma el primer y único elemento del xml que lleva el control del total de usuarios y le decremento uno
        foreach ($xml as $total) {
            $suma = $xml->totalOfUsers;
            $suma = $suma - 1;
            $xml->totalOfUsers = $suma;
        }
        # para evitar que el contador pueda ser negativo, lo dejo en 0
        if($total->totalOfUsers<0) {
            $total->totalOfUsers=0;
        }
        # guardo el nuevo xml
        $xml->asXML('../xml/UserCounter.xml');
    }
    echo "<script>document.location.href='Layout.php';</script>";
?>
```

Al igual que en el opcional 1, creo un div con un id llamado **ucounter** (en referencia a user counter). Como me ha dado problemas para actualizar ambos divs en dos archivos js distintos, lo he resuelto poniendo ambas consultas dentro de la misma consulta AJAX. No es muy prolijo que quede dentro de un archivo que se llama **CountQuestions.js**, pero al menos funciona bien ahora[1] y actualiza cada dos segundos el número de usuarios conectados.

```
setInterval(function() {

        /* creamos la variable con la solicitud */
        xhr = new XMLHttpRequest();

        /* indico qué traer y le paso el parámetro de email para saber las preguntas del usuario */
        xhr.open('GET','../php/CountXMLQuestions.php?email='+$("#email").val(), true);

        /* le digo qué hacer cuando llegue la respuesta */
        xhr.onreadystatechange = function(){
            if(xhr.readyState == 4 && xhr.status == 200) {
                $("#qcounter").html(xhr.responseText);
                //document.getElementById('qcounter').innerHTML = xhr.responseText;
            }
        }
        xhr.send('');

        /* creamos la variable con la solicitud */
        xhr2 = new XMLHttpRequest();

        /* indico qué traer y le paso el parámetro de email para saber las preguntas del usuario */
        xhr2.open('GET','../php/CountXMLUsers.php', true);

        /* le digo qué hacer cuando llegue la respuesta */
        xhr2.onreadystatechange = function(){
            if(xhr.readyState == 4 && xhr2.status == 200) {
                //$("#ucounter").html(xhr2.responseText);
                document.getElementById('ucounter').innerHTML = xhr2.responseText;
            }
        }
        xhr2.send('');
```

[1] https://es.wikipedia.org/wiki/El_fin_justifica_los_medios

```
}, 2000);
```

Ahora bien, resta vincular el login y el logout con las funcionalidades de incremento y decremento. Para asegurar que pasa por ellos, puse ambos archivos como pasarela entre el login y el layout, y el logout y el layout. De esta manera, me aseguro que accede a incrementar o decrementar el contador del XML, aunque esa página sea transparente para el usuario ya que es sólo código php.

La aplicación funciona correctamente y mantiene el control del número de usuarios simultáneos, siempre y cuando ingresen haciendo login y logout.

Para evitar los primeros segundos de demora en cargar la información de los contenedores, copié el código en la vista HandlingQuizesAjax.php para que se precargue en el primer instante con la información que contengan los XML en ese momento. Luego de esa primera carga, ya tiene tiempo suficiente para seguir actualizándose utilizando AJAX.