Acceso a datos entre ventanas de diferentes dominios

Chrome no permite el acceso o comunicación entre ventanas o iframe de diferentes dominios. Es decir, si nosotros tenemos nuestra web en http://www.midominio.com y desde la web abro una ventana (ventana hija) con la URL http://www.google.com , desde la ventana hija no podre accede a los elementos de la ventana padre (quien nos abrió). Para realizar la comunicación entre ventanas de diferentes dominios se debe configurar a nivel de servidor una relación de confianza entre ellos o bien hacer uso del método "**postMessage**()" (https://developer.mozilla.org/en-US/docs/Web/API/Window/postMessage).

Para nosotros poder realizar las pruebas en "local" vamos a tener problemas con el Chrome y para evitarlo les recomiendo que permitan el acceso a los ficheros locales. Se trata de iniciar el Chrome con una directiva " **--allow-file-access-from-files**". De esta forma, ya podremos realizar las pruebas en local y veremos como funcionan todos los ejemplos y ejercicios. Para saber cómo realizar esta parametrización os recomiendo que visiten el siguiente enlace https://www.chromium.org/for-testers/command-line-flags.

CONTROL DE VENTANAS SECUNDARIAS CON JAVASCRIPT

Una de las utilidades más interesantes de JavaScript es el control de ventanas secundarias (popups en inglés). Las ventanas secundarias son las que se abren adicionalmente a la ventana principal del navegador. Suelen ser más pequeñas que las ventanas normales y generalmente no tienen los menús del navegador, barra de direcciones, etc.

Con JavaScript podemos controlar los popups para hacer cosas como abrirlos, cerrarlos, darles el foco, pasar información de una ventana a otra, etc. En este manual veremos cómo se hacen todas estas cosas con las ventanas secundarias para aprender a controlarlas perfectamente.

El índice que vamos a tratar con respecto al tema de las ventanas secundarias o popups es el siguiente:

1. Abrir ventanas secundarias
2. Comunicación entre ventanas
    - 2.1.1. Acceso desde ventana principal a ventana secundaria
    - 2.1.2. Acceso desde ventana secundaria a ventana principal
    - 2.1.3. Acceso a variables y funciones de otras ventanas
3. Cerrar Ventanas

ABRIR VENTANAS SECUNDARIAS EN JAVASCRIPT

Para ello utilizamos el método open del objeto window, que es el encargado de abrir la ventana. La sintaxis del método es la siguiente:

```
window.open(URL,nombre_ventana,parametros_configuración)
```

El primer parámetro es la URL de la página que deseamos mostrar en la ventana secundaria.
El segundo parámetro es el nombre que le damos a la ventana. Este nombre es utilizable desde el HTML para que sirva como target de enlaces o formularios. Es decir, si colocamos un enlace y queremos que actualice la ventana secundaria ponemos el atributo TARGET del enlace igual al nombre de la ventana.
El tercer parámetro es la configuración de la ventana, para indicar el tamaño, qué barras de menús se tienen que ver y cuales no, si tiene o no barras de desplazamiento, etc.
A continuación podemos ver un ejemplo de utilización del método open:

```
window.open("pagina.html","miventana","width=300,height=200,menubar=no")
```

Apertura y configuración de popups con JavaScript

En determinadas ocasiones es muy útil abrir un enlace en una ventana secundaria, es decir, una ventana aparte que se abre para mostrar una información específica. Algunas ventajas de abrir un enlace en una

ventana secundaria pueden ser que:

- El usuario no se marcha de la página donde estaba el enlace.
- La ventana secundaria se puede configurar libremente con lo que se pueden hacer ventanas más grandes o pequeñas y con más o menos menús.
- En general, el grado de control de la ventana secundaria utilizando JavaScript aumenta.

Para abrir una ventana secundaria podemos hacerlo de dos maneras, con HTML y con JavaScript. Veamos cada una de ellas.

Abrir una ventana con HTML

Se puede conseguir abrir una ventana secundaria muy fácilmente con tan solo HTML. Para ello podemos utilizar el atributo TARGET de las etiquetas HREF. Si ponemos target="_blank" en el enlace, la página se abrirá en una ventana secundaria. También podemos poner target="xxx" para que el enlace se presente el la ventana llamada xxx o en el frame xxx. El enlace tendría que tener esta forma:

```
<a href="mipagina.html" target="_blank">
```

El problema de abrir una página secundaria con HTML consiste en que no podemos definir la forma de ésta ni podremos ejercer mayor control sobre ella, tal como comentábamos entre las ventajas de abrir una ventana secundaria con JavaScript. La ventana que se abre siempre será como el usuario tenga definido por defecto en su navegador.

Abrir una ventana con Javascript

Para abrir una ventana con JavaScript podemos utilizar la sentencia window.open(). No pasa nada por que no conozcas JavaScript, puesto que es muy sencillo utilizarlo para este caso. Vamos a ver paso a paso cómo abrir una ventana secundaria utilizando JavaScript.

Sentencia JavaScript para abrir una ventana

La sentencia es simplemente la función window.open(), lo más complicado es saber cómo utilizar esa función, pero ahora veremos que no reviste ninguna complicación.

La función window.open() recibe tres parámetros, que se colocan dentro de los paréntesis, de este modo:

```
window.open(URL,nombre_de_la_ventana,forma_de_la_ventana)
```

Veamos rápidamente cada uno de estos parámetros por separado.

URL: representa el URL que deseamos abrir en la ventana secundaria, por ejemplo http://www.google.com

nombre_de_la_ventana: es el nombre que se le asigna a esta ventana para dirigir enlaces con el atributo target del HTML

forma_de_la_ventana: se indica el aspecto que va a tener la ventana secundaria. Por ejemplo se puede definir su altura, anchura, si tiene barras de desplazamiento, etc

Veamos un ejemplo de sentencia Javascript completa para abrir una ventana secundaria:

```
window.open("http://www.google.com","ventana1","width=120,height=300,scrollbars=NO")
```

Esto quiere decir que abra la página inicial de desarrolloweb.com en una ventana secundaria a la que vamos a llamar ventana1. Además, la ventana será de 120 pixels de ancho, 300 de alto y no tendrá barras de desplazamiento.

Una aclaración adicional, si después de abrir esa ventana colocamos otro enlace en la página que abría la ventana cuyo atributo target está dirigido hacia el nombre_de_la_ventana (en este caso ventana1), este enlace se mostrará en la ventana secundaria.

Función que abre una ventana

Lo más cómodo para abrir una ventana es colocar una función JavaScript que se encargue de las tareas de abrirla y que reciba por parámetro la URL que se desea abrir.

El script es sencillo, veámoslo a continuación:

```
<script type=”text/javascript”> function
ventanaSecundaria (URL){
    window.open(URL,"ventana1","width=120,height=300,scrollbars=NO");
}
</script>
```

Colocamos un enlace

Este enlace no debe estar dirigido directamente a la página que queramos abrir, sino a la sentencia JavaScript necesaria para abrir la ventana secundaria. Para ejecutar una sentencia JavaScript con la pulsación de un enlace lo hacemos así:

```
<ahref="javascript:sentencia_javascript_para_abrir_la_ventana">
```

El enlace llama a la función que abre la ventana

Ahora Veamos cómo quedaría todo ese enlace en la página.

```
<a href="javascript:ventanaSecundaria('http://www.google.com')"> Pincha en este enlace para abrir la ventana secundaria</a>
```

Hay que fijarse que las comillas simples que se colocan para definir el URL que se pasa como parámetro de la función ventanaSecundaria(). Son comillas simples porque el href del enlace ya tiene unas comillas dobles, y dentro de comillas dobles siempre se han de utilizar comillas simples a no ser que deseemos cerrar las comillas dobles.

Parámetros para dar forma a una ventana

Estos atributos los puedes utilizar en la función window.open() para definir la forma que deseas que tenga tu ventana secundaria.

Width	Ajusta el ancho de la ventana. En pixels
Height	Ajusta el alto de la ventana
Top	Indica la posición de la ventana. En concreto es la distancia en pixels que existe entre el borde superior de la pantalla y el borde superior de la ventana.
Left	Indica la posición de la ventana. En concreto es la distancia en pixels que existe entre el borde izquierdo de la pantalla y el borde izquierdo de la ventana.
Scrollbars	Para definir de forma exacta si salen o no las barras de desplazamiento. scrollbars=NO hace que nunca salgan. Scrollbars=YES hace que salgan (siempre en ie y solo si son necesarias en NTS).
Resizable	Establece si se puede o no modificar el tamaño de la ventana. Con resizable=YES se puede modificar el tamaño y con resizable=NO se consigue un tamaño fijo.
Directories (barra directorios)	A partir de aquí se enumeran otra serie de propiedades que sirven para mostrar o no un elemento de la barra de navegación que tienen los navegadores más populares, como podría ser la barra de menús o la barra de estado. Cuando ponemos el atributo=YES estamos forzando a que ese elemento se vea. Cuando ponemos atributo=NO lo que hacemos es evitar que ese elemento se vea.
Location (barra direcciones)	
Menubar (barra de menús)	
Status (barra de estado)	
Titlebar (la barra del título)	
Toolbar (barra de	

Abrir una ventana sin un enlace

En otras ocasiones desearemos abrir una ventana secundaria automáticamente, es decir, sin necesidad de que el usuario pulse sobre ningún enlace. En este caso, el código de la función ventanaSecundaria nos sirve también y habrá que añadir una línea de código JavaScript a continuación de la función ventanaSecundaria. Esta línea a añadir simplemente será una llamada a la función que se ejecutará según se está cargando la página. Veamos como quedaría este código:

```
<script type=”text/javascript”> function
ventanaSecundaria (URL){
    window.open(URL,"ventana1","width=120,height=300,scrollbars=NO");
}

ventanaSecundaria("http://www.google.com");
</script>
```

Queda en negrita lo que sería la llamada a la función que abre la ventana secundaria, como está fuera de una función se ejecuta según se está cargando la página.

COMUNICACIÓN ENTRE VENTANAS

La gracia del trabajo con ventanas secundarias consiste en que tanto la ventana principal como el popup se puedan comunicar entre si y mandarse órdenes y comandos desde una a la otra.
La comunicación podrá ser en dos sentidos:

- Desde la ventana principal a la secundaria.
- Desde la ventana secundaria a la principal.

Si queremos comunicar desde la ventana principal hacia la secundaria necesitamos disponer de una referencia de dicha ventana secundaria o popup. La referencia la será el nombre que le pongamos a la ventana secundaria. A continuación en este artículo veremos cómo asignar un nombre a un popup.
Si la comunicación es desde la ventana secundaria a la principal debemos utilizar el atributo opener de dicha ventana secundaria, que referencia a la ventana principal. También veremos más adelante este tipo de comunicación.

Nombre de la ventana con JavaScript

Cuando abrimos una ventana utilizando el método open del objeto window asignamos un nombre a la ventana para referirnos a ella utilizando HTML. Pero si queremos referirnos a ella utilizando JavaScript necesitaremos utilizar otro nombre.
La referencia JavaScript a la ventana que se acaba de abrir se obtiene gracias al valor de retorno del método open. Para guardar la referencia asignamos el valor de retorno del método a una variable. A partir de ese momento la variable será un sinónimo del objeto window, es decir, será como si fuera el objeto window del popup y por lo tanto podremos acceder a los métodos y propiedades de la ventana secundaria a partir de esa variable.

```
referenciaVentana =window.open(“mi_url.html”,”nombre”,”width=100,height=300”);
```

Luego podremos acceder a los métodos y propiedades de esta manera.

```
referenciaVentana.close();
referenciaVentana.document.bgColor = “red”;
```

EJEMPLO DE ACCESO A UN POPUP

Vamos a hacer un ejemplo de acceso a una propiedad del objeto window de la ventana secundaria. Se trata de la propiedad location del objeto window, que no hemos visto todavía en ejemplos. Recordemos que esta propiedad contiene el URL del documento que estamos viendo y que si la cambiamos cambia la página que se está viendo.

Nuestro ejemplo consiste en una página que va a abrir la ventana secundaria. Además la página tendrá un formulario con un campo de texto y un botón. En el campo de texto podremos colocar URLs y al pulsar el botón haremos que en el popup se muestre la URL que se haya escrito.

Al abrir la ventana guardamos la referencia en la variable miPopup.

```
var miPopup;
miPopup =window.open("about:blank","miventana","width=600,height=400,menubar=no");
```

Si nos fijamos veremos que el URL de inicio de la ventana es about:blank, esto quiere decir que la ventana secundaria se inicializará con un documento en blanco. (Si escribimos about:blank en la barra de direcciones de un navegador veremos que la página se pone en blanco)

Ahora vemos el formulario que contiene el campo de texto y el botón.

```
<form name=formul>
        <input type=text name=url size=30 value="http://">
        <input type=button value="Mostrar URL en popup" onclick="muestraURL()">
</form>
```

No tiene nada que destacar, de modo que pasamos a ver la función que se encarga de actualizar la URL de la ventana secundaria.

```
function muestraURL(){
    miPopup.location = document.formul.url.value;
}
```

La función es extremadamente simple. Sólo se accede a la propiedad location de la variable que guarda la referencia JavaScript y se actualiza con lo que haya escrito en el campo de texto. La propiedad location de la variable es como la propiedad location de la ventana secundaria.
Hay un detalle poco agradable en este ejemplo. Se trata de que al trabajar sobre el formulario de la página que abrimos en primer lugar y luego actualizar el contenido del popup, se queda en segundo plano el popup con el contenido actualizado, tapado por la primera ventana. Esto es porque el foco de la aplicación lo tiene la ventana original, y por eso es la ventana que se ve delante. Sería interesante que al actualizar el contenido del popup se mostrase delante la ventana del popup actualizada. Esto se consigue utilizando el método focus() del objeto window, que coloca el foco de la aplicación en la ventana sobre la que lo invoquemos.
La función con esta pequeña modificación quedará así.

```
function muestraURL(){
    miPopup.location = ocument.formul.url.value;
    miPopup.focus();
}
```

Sólo hemos añadido la llamada al método focus() del objeto window correspondiente al popup.

ACCESO DESDE EL POPUP A LA VENTANA PRINCIPAL

También podemos acceder desde un popup a la ventana que lo abrió, para acceder a los métodos y propiedades de la ventana, o hacer lo que deseemos con JavaScript.
Para ello, en el popup hay una variable declarada que es opener. En realidad es una propiedad del objeto window del popup, que hace referencia al objeto window de la ventana "abridora" (opener).
Para ilustrar el funcionamiento de opener vamos a hacer un ejemplo que consiste en una página principal, que tiene un formulario donde, entre otros campos, debemos introducir un teléfono con su prefijo internacional. En caso de que no conozcamos los prefijos internacionales nos permite pulsar sobre un botón y mostrar una ayuda. La ayuda se ofrece a través de un popup que contiene la lista de los prefijos internacionales para que seleccionemos uno de ellos. En el momento que se selecciona se debe actualizar el formulario de la ventana padre y cerrar la ventana secundaria. Ahora vamos a echar un vistazo a la página

principal, que contiene el formulario donde se debe escribir el prefijo y el número. Al lado del campo de texto donde se coloca el prefijo hemos puesto un botón que sirve para mostrar el popup de ayuda. Al pulsar el botón llamamos a la función abreVentana() que será la encargada de abrir el popup con la lista de prefijos.

```
<html>
<head>
	<title>Formulario prefijos</title>
<script>
var miPopup;
function abreVentana(){
  miPopup = window.open("prefijos.html","miwin","width=300,height=150,scrollbars=yes");
  miPopup.focus();
}

</script>
</head>
<body>
<form name=formul>
<table cellspacing="3" cellpadding="3" border="0">
<tr>
	<td align="center">Prefijo</td>
	<td align="center">Número</td>
</tr>
<tr>
	<td align="center">
		<input type=text name=prefijo value="+" size=3 maxlength=3>
		<input type="Button" value="?" onclick="abreVentana()">
	</td>
	<td align="center">
		<input type=text name=numero value="" size=10 maxlength=10>
	</td>
</tr>
</table>
</form>
</body>
</html>
```

La función abreVentana se tiene que entender perfectamente. El resto de la página es un simple formulario HTML metido dentro de una tabla para que quede mejor maquetado.
Ahora veamos la página HTML del popup, que es la que utiliza la referencia opener a la ventana "abridora", para actualizar el campo del formulario donde hay que colocar el prefijo.

```
<html>
<head>
	<title>Prefijos internacionales</title>
<script>
function ponPrefijo(pref){ opener.document.formul.prefijo.value =
	pref; window.close();
}
</script>
</head>

<body>
<h1>Lista de prefijos internacionales</h1>
<form name=fprefijos>
España:
<input type="Button" value="+34" onclick="ponPrefijo('+34')">
<br>
Holanda:
<input type="Button" value="+31" onclick="ponPrefijo('+31')">
<br>
Gran Bretaña:
```

```
<input type="Button" value="+44" onclick="ponPrefijo('+44')">
<br>
Estados Unidos:
<input type="Button" value="+01" onclick="ponPrefijo('+01')">
<br>
Bélgica:
<input type="Button" value="+32" onclick="ponPrefijo('+32')">
<br>
Grecia:
<input type="Button" value="+30" onclick="ponPrefijo('+30')">
</form>
</body>
</html>
```

Tenemos un formulario con una serie de botones para seleccionar el prefijo. Todos llaman a la función ponPrefijo() pasándole el prefijo seleccionado.
La función ponPrefijo() debe poner en un campo del formulario de la ventana que abrió el popup el valor del prefijo que ha recibido por parámetro. Para ello accede a la propiedad opener del objeto window, para tener acceso a la ventana principal (abridora) y desde allí accede a su documento, al formulario y a la propiedad value del campo de texto donde hay que escribir el prefijo. Por último cierra el popup con el método close() del objetowindow.

ACCESO A VARIABLES Y FUNCIONES DE OTRAS VENTANAS

Desde una ventana también tenemos acceso a las variables y funciones que hayamos declarado en otras ventanas. Gracias a esto, desde un popup podemos controlar el estado de las variables de la página principal y llamar a funciones para hacer cualquier cosa que necesitemos.
El acceso a las variables y las funciones se realiza a través de los objetos ventana. Por ejemplo, si deseamos desde una ventana tener acceso a una función de un popup colocaríamos su objeto ventana seguido de un punto y el nombre de la función a la que queramos acceder, como si fuese un método nuevo del objeto ventana. Las variables se acceden como si fueran propiedades del objeto ventana. El código sería algo parecido a esto.

Para acceder a una variable miPopup.miVariable
Para acceder a una función miPopup.miFunción()

Vamos a ver un ejemplo para ilustrarlo. Tenemos una página con un número cualquiera de campos de texto en un formulario y una función que sirve para inicializar los datos de los campos de texto a 0. En esta página abriremos una ventana secundaria desde la cual invocaremos a la función que inicializa los campos de texto.

La página principal tendrá una parte con un script para definir la función y abrir el popup y otra parte con el formulario que contiene los campos de texto. Se puede ver a continuación su código.

```
<html>
<head>
<title>Inicializador de formularios</title>
<script type=”text/javascript”> function
inicializaCampos(){
        for (i=0;i<document.forms[0].elements.length;i++)
                document.forms[0].elements[i].value = "0";
        }
        var miPopup;
        miPopup = window.open("llamadas-desde-ventanas-popup.html","miventana","width=308,height=70,menubar=no");
</script>
</head>

<body>
<form>
```

```
        <input type="Text" name="t1" value="0" size="4" maxlength="100">
        <input type="Text" name="t2" value="0" size="4" maxlength="100">
        <input type="Text" name="t3" value="0" size="4" maxlength="100">
        <input type="Text" name="t4" value="0" size="4" maxlength="100">
        <input type="Text" name="t5" value="0" size="4" maxlength="100">
        <input type="Text" name="t6" value="0" size="4" maxlength="100">
        <input type="Text" name="t7" value="0" size="4" maxlength="100">
</form>
</body>
</html>
```

En el popup podremos encontrar el código necesario para invocar a la función inicializaCampos() que pertenece a la ventana principal. La llamada a la función de la ventana "abridora" se realiza al pulsar un botón.

```
<html>
<head>
<title>Popup Inicializador de formularios</title>
<script>
function llamadaOtraVentana(){
//llamada a la función de ventana abridora
        window.opener.inicializaCampos();
        window.opener.focus();
}
</script>
</head>
<body>
<form>
        <input type="button" value="Llamar a función de otraventana" onClick="llamadaOtraVentana()">
</form>
</body>
</html>
```

CERRAR VENTANAS CON JAVASCRIPT

Para cerrar la ventana del navegador debemos utilizar el método close() del objeto window de la ventana que deseamos cerrar. Es muy sencillo, aun así, vamos a ver un ejemplo sobre cerrar ventanas.
El ejemplo consta de una página que tiene un botón que abre una ventana secundaria o popup. Además, tendrá otro botón que cerrará la ventana secundaria. Por su parte, la ventana secundaria tendrá un botón que permitirá cerrarse a ella misma. Luego de tratar este ejemplo comentaremos el caso especial de cerrar la ventana principal desde el popup y las pegas que tiene.

La página principal tendría esta forma:

```
<html>
<head>
<title>Ventana Principal</title>
<script>
//creamos la variable ventana_secundaria que contendrá una referencia al popup que vamos aabrir
//la creamos como variable global para poder acceder a ella desde las distintasfunciones
var ventana_secundaria;

function abrirVentana(){
//guardo la referencia de la ventana para poder utilizarla luego
  ventana_secundaria = window.open("cerrar_window2.html","miventana","width=300,height=200,menubar=no");
}

function cerrarVentana(){
//la referencia de la ventana es el objeto window del popup. Lo utilizo para acceder al método close
   ventana_secundaria.close();
}
```

```
</script>
</head>

<body>
Esta es la ventana principal
<form>
<input type=button value="Abrir ventana secundaria"onclick="abrirVentana()">
<br>
<br>
<input type=button value="Cerrar ventana secundaria"onclick="cerrarVentana()">
</form>

</body>
</html>
```

Contiene un script en la cabecera de la página con las funciones para abrir y cerrar la ventana. Además, como ya se indicó anteriormente en este manual, se debe guardar una referencia a la ventana que se acaba de abrir para poder realizar acciones sobre ella posteriormente, en este caso cerrarla. Como la referencia de la ventana se crea en la función abrirVentana() y luego se utilizará en la función cerrarVentana(), dicha referencia deberá declararse como global o de lo contrario sólo tendría validez mientras la función abrirVentana() estuviera ejecutándose y, cuando pretendiésemos utilizarla en la función cerrarVentana(), nos diría que esa referencia ya no existe.
Así que disponemos de una referencia global a la ventana que va a abrirse y dos funciones que abren el popup guardando la referencia y cerrar el popup utilizando la referencia de la ventana a cerrar. Estas dos funciones se llaman desde dos botones de la página, que hemos colocado dentro de un formulario.

> Nota: Hay que señalar que puede haber un error JavaScript si se pretende cerrar la ventana antes de abrirla. Para evitarlo podemos controlar que realmente la ventana está abierta antes de ejecutar el método close() de su objeto window.

Por su parte, el popup tiene un código como el siguiente.

```
<html>
<head>
<title>Ventana Secundaria</title>
<script>
function cerrarse(){
        window.close();
}
</script>
</head>

<body>
Esta es la ventana del popup
<form>
        <input type=button value="Cerrar"onclick="cerrarse()">
</form>

</body>
</html>
```

Contiene tan solo una función para cerrarse ella misma, que lo único que hace es ejecutar el método close() sobre su propio objeto window. También hemos colocado un botón que se ha configurado para llamar a la función que cierra la ventana cuando se le haga clic.

Un detalle sobre cerrar ventanas

En cualquier momento podemos cerrar una ventana secundaria utilizando el método close() del objeto

window. En el ejemplo anterior hemos visto cómo se cierran ventanas y es muy sencillo. En cualquier momento podemos cerrar una ventana, pero si intentamos cerrar una ventana que no se ha abierto con JavaScript (sin utilizar window.open()) nos saldrá una caja de confirmación que pregunta al usuario si de verdad quiere cerrar la ventana.

Podemos ver esta caja de confirmación al pulsar el botón siguiente, que intentará cerrar esta ventana.

> Nota: La caja de confirmación que hemos podido ver aparece como elemento de seguridad, para evitar que un programador malicioso intente cerrar una ventana que hemos abierto nosotros como usuarios y que, en teoría, no tiene permiso para cerrar. Así que a partir de determinado navegador decidieron preguntar al usuario si realmente desea que se cierre esa ventana.

Es sólo un detalle que no tiene mucha importancia, pero para las personas que nunca han experimentado con el trabajo con ventanas secundarias en JavaScript, resultará sorprendente que el navegador haga esa pregunta, que probablemente nunca hayamos visto. Una cosa más, sólo ocurre esto en navegadores modernos, aunque actualmente le va a pasar a casi todos los usuarios.

SUBMENÚ EN OTRA VENTANA

Vamos a realizar un pequeño ejemplo sobre cómo trabajar con ventanas secundarias o popups. Las ventanas secundarias son esas ventanitas que se abren a parte de la ventana principal del navegador y por lo general molestan un poco en determinados sitios gratuitos.

Para abrir esas ventanas se utiliza el lenguaje JavaScript, más concretamente, el método open() del objeto window. En este ejemplo vamos a ir un poco más allá, vamos a crear una ventana secundaria y desde ella vamos a acceder a las propiedades de la ventana padre, es decir, la ventana que la abrió. El ejercicio será el siguiente:

Tendremos una página con fondo blanco, un campo de texto vacío y un botón. Cuando se pulse el botón abriremos un popup que contendrá una tabla con los colores puros de HTML. El visitante podrá seleccionar uno de esos colores y entonces el fondo de la página padre cambiará a ese color y el color se escribirá en el campo de texto.

Página padre

El siguiente código debe probarse en un servidor Web ya que por motivos de seguridad algunos navegadores dan problemas a la hora de acceder al opener.
La página original contendrá un simple formulario con un botón y un campo de texto. Además, contendrá el script JavaScript necesario para abrir la ventana secundaria.

```
<html>
<head>
<title>Submenú en ventana a parte</title>
<script type="text/javascript"> function
lanzarSubmenu(){
      window.open("submenu_ventana2.html","ventana1","width=400,height=400,scrollbars=YES");
}
</script>
</head>
<body bgcolor="#ffffff">
<form>
<input type="text" name="colorin" size="12" maxlength="12">
<br><br>
<input type="button" value="Lanzar submenu" onclick="lanzarSubmenu()">
</form>
</body></html>
```

La función lanzarSubmenu() es la que contiene el script para abrir el popup.

El formulario es de lo más normal. Lo único destacable es el atributo onclick del botón, que sirve para definir las acciones a ejecutar cuando se pulsa el botón, en este caso una llamada a la función que abre la ventana secundaria.

Página secundaria

La página secundaria es un poco más compleja, pero la parte que nos importa a nosotros en este ejercicio es muy sencilla. Lo importante de la página es que tiene que acceder a la ventana padre para modificar su estado y para ello utiliza un objeto JavaScript: opener.

El objeto opener está disponible en las páginas que son ventanas secundarias y hace referencia a la ventana que la abrió, es decir, la ventana padre. Dicho de otro modo, el objeto opener en la ventana popup es un sinónimo del objeto window en la ventana padre.

El script que accede a la ventana padre es el siguiente:

```
<script language="JavaScript"> function
actualizaPadre(miColor){
        window.opener.document.bgColor = miColor; window.opener.document.forms[0].colorin.value =miColor;
}
</script>
```

La función actualizaPadre() es la encargada de realizar el trabajo. Recibe el código del color sobre el que se ha pulsado. En la primera línea cambiamos el color de la página web padre y en la segunda línea colocamos el código RGB del color recibido por parámetro en el campo de texto.
Como vemos, el objeto opener también depende del objeto window de la página, como todos los demás objetos de la jerarquía JavaScript.

```
<table width="80%" align="center" cellpadding="1"cellspacing="1">
<script language="javascript">
var r = new Array("00","33","66","99","CC","FF");
var g = new Array("00","33","66","99","CC","FF");
var b = new Array("00","33","66","99","CC","FF");
for (i=0;i<r.length;i++)
        for (j=0;j<g.length;j++) {
              document.write("<tr>"); for
              (k=0;k<b.length;k++) {
                    var nuevoc = "#" + r[i] + g[j] + b[k];
                    document.write("<td bgcolor=\"" + nuevoc + "\" align=center><font size=1 face=verdana>");
                    document.write("<a href=\"javascript:actualizaPadre('" + nuevoc + "')\">"); document.write(nuevoc);
              }
              document.write("</a></font></tr>");
        }
</script>
</table>
```

Lo importante para nosotros ahora es entender que este script crea una tabla con todos los colores puros, colocados en una celda cada uno. Dentro de cada celda se escribe un enlace que llama a la función actualizaPadre() pasándole el código del color se ha de utilizar.