Memoria Práctica 1

Página web HTML/CSS con menús jQuery

Samuel Martín Gómez-Calcerrada

Juan Antonio Echeverrías Aranda

ÍNDICE

INTRODUCCIÓN

Esta práctica está basada en la página web estática que se propuso como ejercicio en clase, usando HTML y CSS. La teníamos terminada por lo que no nos ha supuesto mucho trabajo cambiar la funcionalidad de los menús que ya tenía con CSS a JQuery.

Para la realización de la página web hemos utilizado Brackets, que es un editor de código abierto para el diseño y desarrollo web construido sobre tecnologías HTML, CSS y JavaScript. Brackets ha sido creado y es mantenido por Adobe y alguna de sus características más relevantes son la posibilidad de abrir directamente el navegador para hacer previsualizaciones de nuestro código y la posibilidad de añadir plugins.

Está página web consta de cuatro archivos dónde se implementan las distintas tecnologías del lado del cliente:

- **index.html:** implementado con el lenguaje HTML, basado en etiquetas, nos proporciona la información estructurada en títulos, secciones, párrafos...
- **main.css:** implementado con el lenguaje CSS, nos permite añadir estilos al documento.
- **main.js:** implementado con el lenguaje JavaScript para añadir interactividad en la página.
- **jquery.js:** biblioteca JavaScript que usamos para gestionar la página.

A continuación procedemos a comentar los elementos más relevantes del diseño de la página.

HTML

En la cabecera indicamos el título de nuestra página mediante la etiqueta ’<title> </title>’, el cual aparecerá en la pestaña y añadimos las etiquetas correspondientes para hacer referencia al resto de archivos.

En el cuerpo está el contenido del documento que se visualizará en el navegador y lo hemos estructurado mediante el contenedor ’div ’ en cinco bloques: una cabecera general, un menú principal, una sección *“home”* para volver a la página principal, el contenido y un *“footer”*. Todos estos bloques están a su vez contenidos en un ’div ’ y constan de atributos ’id ’ y ’class’ para especificar sus propiedades correspondientes en el archivo *main.css*.

Destacamos el bloque del menú principal ’ ’, el cual está implementado como una lista no ordenada y uno de sus elementos es a su vez otra lista no ordenada. El bloque del contenido ‘ ’ contiene dos bloques, uno que consta de un menú implementado mediante una lista no ordenada ‘ ’ y otro que alberga la información de la página ‘ ’. Podemos acceder a las distintas partes del contenido de la información mediante el menú citado anteriormente, esto lo conseguimos poniendo “anclas”, las cuales sirven para definir posiciones en el documento. Las anclas se crean con la etiqueta ’<a> ’ y el atributo ’name ’ (Ej: <a name:”nombre_ancla”>)

CSS

Vamos a comentar solo los aspectos que nos parecen más relevantes.

Lo primero que hacemos es quitar todos los ’margin’ y todos los ’padding ’ de toda la definición por defecto del navegador, poniéndolos a 0, para aplicar nuestros estilos.

Lo que más nos ha costado hacer ha sido el menú principal. Este menú es una lista y para hacer que se muestre horizontalmente aplicamos a los elementos de la lista la propiedad ’float:left’. Al hacer esto nos encontramos con el problema de que al aplicar dicha propiedad, los elementos de la lista se salían de su contenedor y este pasaba a tener un tamaño mínimo. Para solucionar este problema añadimos al contenedor la propiedad ’overflow:hidden’, para obligarle a albergar todo el contenido que tuviese dentro, pero al implementar el submenú desplegable nos

encontramos con la sorpresa de que esté no se mostraba. Después de darle vueltas quitamos la propiedad ’overflow’ y añadimos una etiqueta ’div’ sin contenido después de los elementos posicionados de forma flotante, con la propiedad ’clear:both’, para forzar al contenedor a tener la altura suficiente y de esta forma ya se nos mostraba el submenú desplegable.

El bloque ’ ’ tiene la propiedad ’overflow:hidden’ para albergar a los bloques ’ ’ y ’ ’ que poseen la propiedad ’float:left’.

Por último comentar que el “*footer*”, al encontrarse debajo de elementos flotantes, tiene la propiedad ’clear:both’ para que aparezca debajo de éstos.

JavaScript

Al principio intentamos usar los eventos de JQuery 'FadeIn(FadeOut)' y 'SlideUp(SlideDown)' en el “hover” del menú para que se desplegara un submenú en el caso de que existiera 'ul li:has(ul)'.

Nuestro primer problema fue descubrir que los eventos en JQuery se implementan por defecto en una cola, y teníamos que usar los dos al mismo tiempo, por lo que usamos '.dequeue()' para eliminar la cola.

```
$(".menu-principal ul li ul li").slideDown(2000).fadeIn(2000).queue();
```

Después de varios días intentando explicarnos porque no funcionaban correctamente descubrimos que si se quieren usar estos dos efectos simultáneamente hay que usar en lugar de 'FadeIn(Out)', un 'animate' que cambie la opacidad del elemento CSS en un tiempo determinado.

```
.slideDown(2000).animate({opacity: 1}, {duration: 2000}).dequeue();
```

Nuestro siguiente problema fue que si el usuario movía muy rápido el ratón entrando y saliendo del menú sin dar tiempo a que los eventos terminaran, se producían errores en la visualización de los menús desplegables.

Esto lo solucionamos indicándole que en el “hover”, antes de nada, terminara los eventos en curso y limpiara la cola con el método '.stop' y después que los dejara preparados por si habían quedado visibles por error.

Para terminar hemos implementado otra función que resalta en un gris más oscuro el elemento en el que estamos situado, tanto en el primer menú, como en el menú hijo agregando o quitando una clase CSS al 'div' correspondiente.