Programador Web – Nivel 1

Unidad 6: Desarrollo web multiplataforma

Contacto: info@elearning-total.com
Web: www.elearning-total.com

Indice

Unidad 6: Desarrollo web multiplataforma

Responsive Design

Objetivos

Que el alumno logre:

- Manejar los nuevos elementos incorporados a CSS3 para crear sitios multiplataforma.

Responsive Design

Hoy en día ha aumentado de forma exponencial el uso dispositivos móviles, además de la demanda de contenido para estos, por lo que cada vez es más necesario una web adaptable a los distintos dispositivos.

¿Qué es el Responsive Design?

Esta técnica de diseño web consiste en crear una estructura de una página web que según el tamaño de la pantalla (o ventana) en la que se visualice variará su contenido para que siempre sea visible y cómodo de usar desde computadoras de escritorio, tablets y smartphones, y se puede poner en práctica esta forma de adaptar el contenido a todo tipo de resoluciones con hojas de estilo CSS o con JavaScript (tenemos que tener en cuenta también en que puede haber personas o bots que tengan desactivado JavaScript).

Beneficios

La web se visualizará en todos los dispositivos que usemos correctamente, sin necesidad de hacer zoom y se adaptará a los giros en dispositivos móviles.

Google tiene en cuenta que páginas usan diseños que se adaptan a dispositivos móviles.

Ayudamos a las personas que tienen discapacidades visuales a que puedan usar más fácilmente la web.

¿Cómo implementarlo?

Podemos ver dos opciones, tener una web existente o crearla de cero, si existe una web, dependiendo de su complejidad puede ser relativamente sencillo adaptarla o puede ser una tarea tediosa (sobre todo si es un CMS con una plantilla desastrosa, a nivel de estructura).

Tipos de Responsive Designs

Podemos determinar los siguientes tipos de responsive design:

Adaptándose al ancho

Este tipo de diseño es uno de los más comunes, que se distingue por no mover en exceso los elementos de la web e intentar que se adapten al ancho de pantalla reduciendo el menú (a veces cambiando la disposición), redimensionado las imágenes y poco más.

Cascada de columnas

En este caso tenemos una web con varias columnas que al visualizarse en pantallas estrechas se pondría una debajo de la otra seguidamente.

Reestructuración

Como el propio título indica en esta variante cambiamos la estructura de los elementos disponiéndolos de una forma distinta según el tamaño de ventana y se puede hacer de muchas formas (hay solo depende del diseño que queramos).

Cambio del diseño visual

Esta se podría decir que no es una forma de estructurar el contenido para hacerlo más fácil de manejar fuera del navegador de escritorio, sino que según el tamaño de la ventana adapta el diseño de una forma u otra por motivos estéticos, y suele usarse en las típicas web que dan la información sobre algo de forma rápida y concisa.

@media de CSS3

La regla @media nos permite especificar que cierto conjunto de reglas CSS solo deben aplicarse a cierto tipo de dispositivo.

Así las definiciones dentro del bloque de la regla @media screen { ... } solo serían interpretadas por dispositivos conectados a monitores de computadoras y los de la regla@media projection { ... } solo se aplicaría a proyectores.

CSS3 añade importantes y nuevas capacidades que nos permiten definir conjuntos de estilos dependiendo de propiedades comunes de los dispositivos que acceden a nuestros sitios. Propiedades como el alto y el ancho o la relación de aspecto o el número de colores disponible. Las reglas @media pueden ser utilizadas para adaptar nuestras páginas, no solo para dispositivos comunes, sino para todo tipo de dispositivos que nuestros usuarios utilicen para visitar nuestros sitios.

MEDIA QUERIES

Hasta ahora, si necesitábamos conocer el tamaño actual de la ventana del navegador, debíamos usar JavaScript para recolectar datos de ese tipo desde el navegador y después darle un uso a esos datos a través de la modificación del DOM a través de métodos programados en JavaScript. Aunque dicho método es válido, no es realmente óptimo ni intuitivo.

CSS3 nos aporta las *media queries* que nos proveen de una forma de conocer algunas de las propiedades comunes de los dispositivos que nos visitan que podemos utilizar en nuestros archivos de estilo para construir entornos dependiendo de los mismos **sin ayuda** de JavaScript.

Aunque *media queries* dispone de muchas propiedades diferentes, podemos identificar 5 principales:

- *Aspect-ratio:* Mira las dimensiones relativas del dispositivo expresadas como una relación de aspecto: 16:9 por ejemplo.
- *Width y height:* Mira las dimensiones del área de visualización. Además pueden ser expresadas en valores mínimos y máximos.
- *Orientation:* Mira si el layout es panorámico (el ancho es mayor que el alto) o vertical (el alto es mayor que el ancho). Esto nos permite ajustar los diseños para dispositivos con propiedades de giro de la pantalla como el iPhone, y otros smartphones y los tablets.

- *Resolution:* Mira la densidad de los pixeles en el dispositivo de salida. Esto es especialmente útil cuando queremos aprovecharnos de las ventajas de los dispositivos que tiene una resolución mayor a 72 dpi.
- *Color, Color-index y monochrome*: Encuentran el número de color o bits por color. Esto nos permite crear diseños específicos para dispositivos monocromáticos.

VIEWPORT

Esta meta-etiqueta fue creada en principio por **Apple** para sus dispositivos móviles, pero se ha convertido en todo un estándar que es soportado por la mayoría de los dispositivos móviles (smartphones, tablets y gran parte de móviles de gama media y baja).

Su uso es totalmente necesario, ya que sino el navegador establece el ancho con el que prefiere visualizar una página en lugar de usar el ancho del que dispone (es decir, si la pantalla de nuestro móvil tiene 400px y el navegador detecta que lo óptimo sería visualizarla con 700px así lo hará si no usamos esta meta-etiqueta).

```
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
```

Se pueden usar los siguientes parámetros (separados por comas):

- *Width:* ancho de la página (se puede establecer en píxeles o como device-width y usará el ancho del que dispone).
- *Height:* alto de la página, actúa igual que el width.
- *Initial-scale:* escala o zoom inicial de la página (este y los demás tipos de escala se establecen con valores como 1.0 para no tener zoom o 2.5 para tener un zoom del 2,5 de aumento, por ejemplo).
- *Minimum-scale:* zoom mínimo que podemos hacer en la página.
- *Maximum-scale:* zoom máximo que podemos hacer en la página.
- *User-scalable:* establece si está permitido o no hacer zoom (yes/no).

La propiedad @media

La propiedad `@media` permite que incluyamos *media queries* directamente en nuestras hojas de estilo. De esta manera **mejoramos el rendimiento** de la página al no tener que cargar archivos CSS adicionales para los diferentes dispositivos.

Para utilizar las reglas **@media** lo único que tenemos que hacer es crear nuestra hoja de estilos y cuando queramos definir un conjunto de reglas de estilo específicas para un dispositivo determinado utilizar la sintaxis:

@media screen and (parámetros) and (parámetros) { ... }

Ejemplo:

Vamos a poner un ejemplo que podría ser funcional, que se basa en un menú que hay en una cabecera y consta de cinco links que mandan a las distintas partes de la web y actuará de la siguiente forma los elementos `<a>`:

- En principio tendrá el siguiente código, que mantendrá a los cinco elementos en línea.

```
a.menu {
        display:inline-block;
        padding:0px 12px;
        margin:0px 8px;
}
```

- Cuando el ancho de pantalla sea inferior a 1200 píxeles (lo que podría ser el ancho máximo de la página) los elementos del título se juntan para que no sobresalgan de la cabecera.

```
@media all and (max-width: 1200px) {
        a.menu {
                display: inline-block;
                padding:0px 6px;
                margin:0px 4px;
        }
}
```

- Cuando el ancho de ventana sea inferior a 840 píxeles se reestructurarán los elementos y pasaran de estar en línea a estar en cascada uno debajo del otro y tomarán un pequeño margen a su izquierda.

```
@media all and (max-width: 840px) {
        a.menu{
                margin:0px;
                padding:0px;
                padding-left:5%;
                display:block;
                float:none;
                text-align:left;
        }
}
```

- Y por último a los 520 píxeles imaginamos que para que se adapte a anchos estrechos el resto de la página ha cambiado de estructura, por lo que ahora dispone el menú de todo el ancho y lo añadiremos un margen izquierdo superior.

```
@media all and (max-width: 520px) {
        a.menu{
                padding-left:20%;
        }
}
```

USANDO MEDIA QUERIES PARA ESPECIFICAR ESTILOS DE UN SITIO

Antes de nada debemos crear nuestra hoja de estilos. Creamos un estilo por defecto que incorporará todas las necesidades **universales** para nuestro diseño y lo guardamos. Estos son los estilos que utilizará para nuestra vista por defecto, habitualmente la vista de escritorio:

```
/**
 * Estilo por defecto
 */

body {
   background: white url(../media/fondo.jpg) no-repeat 0 0;
   margin: 0;
}

header, section, footer {
    width: 960px;
    margin-left: auto;
    margin-right: auto;
}

h1 {
   color: black;
   font-style: italic;
}

h2 {
   color: #717171;
}

p {
   font: normal 100%/1.5 Helvetica, Arial, Sans-serif;
   color: #313131;
}
```

Creamos reglas @media para cada tipo de dispositivo o media para el que queramos diseñar un estilo específico. Por ejemplo, podemos incluir un estilo específico para dispositivos medios, entre 600px y 1023px:

```
/**
 * Estilo para 600 a 1023px
 */
@media all and (min-width:600px) and (max-width: 1023px) {
        header, section, footer {
            width: 580px;
        }

        h1 {
           color: #818181;
        }

        p {
           font: normal 12pt/2 Constantia, palatino times, "times new roman", serif;
           color: #000000;
        }
}
```

Ya tenemos un estilo por defecto y otro para cuando la página tiene un ancho de entre 600px y 1023px. Vamos a añadir otro para el dispositivos más pequeños (celulares), que tengan un ancho menor a 599px:

```
/**
 * Estilo para dispositivos de menos de -599px
 */
@media all and (max-width: 599px) {
header, section, footer {
    width: 90%;
}

h1 {
   color: rgb(140, 120, 120);
   text-shadow: 0 0 5px rgb(0, 0, 0);
}

p {
   font: normal 1em/1.25em "helvetia neue", Helvetia, Arial, Sans-serif;
```

```
    color: rgb(255, 255, 255);
}
```

Ahora vamos a añadir la etiqueta meta de la vista en el head del documento HTML:

Eso **previene** de que dispositivos con una pantalla más pequeña redimensionen la página impidiendo que los estilos se interpreten. Enlazamos nuestro estilo por defecto en el documento HTML a través de la etiqueta **link** y utilizando como regla **media all**.

El código completo del documento HTML sería como el que sigue:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width; initial-scale=1.0; maximum-scale=1.0; user-scalable=0;">
    <title>Ejemplo de Media Queries</title>
    <link rel="stylesheet" media="all" href="/css/default.css">

  </head>
  <body>
   <header>  <h1>La Comunidad del Anillo</h1></header>
    <section>
          <h2><strong>Capítulo I.</strong> Gandalf el Mago</h2>
      <p>Cuando Frodo Bolson de Pipas vio acercarse por el camino el carruaje de
      Gandalf el Mago no pudo evitar soltar un gritito de alegría como
      <em>¡Vaaamono preeeehmoh!</em></p>
    </section>
    <footer>
      <p class="author">by J.R.R. Tolkien </p>
    </footer>
  </body>
</html>
```

Resumen

En esta Unidad…

En la presente unidad desarrollamos los conceptos necesarios para incorporar los atributos gráficos a nuestras estructuras de HTML utilizando el lenguaje CSS3

Con las propiedades propuestas podemos utilizar cualquier familia tipográfica para nuestra web. También trabajamos con la posibilidad de desarrollar sitios adaptables a los diferentes dispositivos disponibles hoy en el mercado.

En la próxima Unidad…

En la próxima unidad vamos comenzar a trabajar con los conceptos de programación, para prepararnos para incorporar contenido dinámico a nuestros sitios.