Diseño gráfico orientado a la Web

Modulo: Diseño de Pagina Web

MSc. Jorge Centeno.

PRÁCTICA : CREAR UNA PLANTILLA HTML DE DISEÑO SENSIBLE

Universidad Nacional Autónoma de Nicaragua, León

CONTENIDO

PRÁCTICA 3: CREAR UNA PLANTILLA HTML DE DISEÑO SENSIBLE (RESPONSIVE DESIGN)

OBJETIVOS

- Crear un plantilla web utilizando los elementos de HTML5
- Aprender los conceptos básicos de diseño sensible o responsive design

DISEÑO WEB SENSIBLE, EL NOMBRE DEFINITIVO.

¿Cómo se dice, responsive web design o diseño adaptativo? Wikipedia lo llama diseño web adaptativo o adaptable. Los usuarios lo llaman diseño web responsive. Google Translate lo llama "diseño sensible" y sin duda nos parece la mejor adaptación. Utilices el nombre que utilices te estás refiriendo a lo mismo, un tipo de diseño web que se adapta a todas las resoluciones de pantalla.

¿Cómo saber si una web tiene diseño responsive?

Identificar si una web es o no es sensible es muy sencillo: sólo tienes que redimensionar la ventana de tu navegador para hacerla más pequeña. Si aparece la barra horizontal de scroll y tienes que desplazarte horizontalmente para ver el contenido oculto entonces… la web no tiene responsive design. Sencillo, ¿verdad?

Esta prueba tan sencilla despeja las dudas entre diseño líquido y diseño.

¿Cuáles son las ventajas del diseño responsive?

El diseño responsive proporciona 4 grandes ventajas.

- SEO. Search Engine Optimization. La optimización para buscadores es la gran ganadora. Sólo hay una web (HTML + CSS) y por tanto la URL es única para todos los dispositivos. Los buscadores no van a encontrar otras URL y contenidos diferentes para los dispositivos móviles. Nos evitamos instrucciones en robots.txt y redirecciones de unas versiones a otras, e incluso evitamos perder los links que podrían hacer los usuarios en determinadas versiones de las páginas.
- UX. User eXperience. El usuario accede al mismo website con independencia del dispositivo que utilice. La versión para dispositivos móviles es también completa. La misma experiencia de uso, el mismo menú de navegación. Coherencia.
- Integridad. Sólo hay una estructura HTML y una única hoja de estilo CSS para todos los dispositivos. Tengan el tamaño que tengan. Desde los smartphones a 320 px de ancho hasta resoluciones de 2.650 px (superiores incluso). Antes esto no era así y había una hoja de estilo para cada ancho estándar (320 px para móviles, 768 px para tablets, 1.024 px para notebooks y 1.600 px para ordenadores de escritorio). Cada actualización en la web suponía retocar una por una las CSS y eso podía representar perdidas de integridad.

- Costes. Aunque el diseño responsive requiere más inversión en tiempo del diseñador y programador, se optimizan los costes de mantenimiento ya que sólo hay que revisar 1 hoja de estilo y no como antes que había que manipular 4 hojas de estilo diferentes

¿Cómo se adapta el contenido de una web con diseño responsive?

La estructura de la página web se define con HTML5. La presentación de la página (color de fondo, tipografías, cuerpos de la fuentes, etc) se hace programando las hojas de estilo CSS3. Cuando el navegador del usuario pide una página, el servidor hace que la hoja de estilo ejecute la MEDIA-QUERY por la que obtiene información inmediata sobre el dispositivo, especialmente la resolución de pantalla y orientación (vertical o PORTRAIT, horizontal o LANDSCAPE). El servidor sólo tiene entonces que ajustar el contenido que va a servir al formato de pantalla en el que va a ser visualizado. Este proceso dura milésimas de segundo, no hay delay (retraso) para el usuario. A todos los efectos es una página más.

Para que esto sea posible debemos definir con antelación la disposición, orden y tamaño que tendrán los diferentes elementos que constituyen el contenido de la página web y también el comportamiento que van a tener esos elementos en tamaño reducido.

El diseño responsive es por esta razón mucho más trabajoso que un diseño convencional ya que implica tener en consideración varias visualizaciones (smartphone, tablet, ordenador de escritorio), reordenar los elementos (que se muestra antes y qué se muestra después) y rediseñar los elementos que puedan plantear problemas (los campos de un formulario de contacto, por ejemplo).

Herramientas online gratuitas para identificar y validar diseño responsive

Para validar el diseño adaptativo de una web tendrás que hacer algunas pruebas. Existen varias herramientas online gratuitas que te facilitarán la labor. Cada día hay más en la red. Las puedes buscar como **"responsive web design testing tools"**. Te dejamos aquí las principales que han sido creadas para atraer tráfico y generar notoriedad.

- Appex (una agencia noruega de publicidad)
- Matt Kersley (diseñador y programador)
- Responsinator (también puede funcionar como bookmarklet)
- Responsive Pixel
- Responsive.is
- Screenfly (ESTA ES LA MEJOR)

BIBLIOGRAFÍA

Niederst Robbins, Jennifer. **Learning Web Design**. Fourth Edition.

Diseño web sensible, responsive web design o diseño adaptativo. http://visibilidad-trafico-conversion.com/2013/03/22/diseno-web-sensible-responsive-web-design-adaptativo/

Diseño web adaptable, práctico. http://www.adinteractive.co/web/diseno-web-adaptable-practico

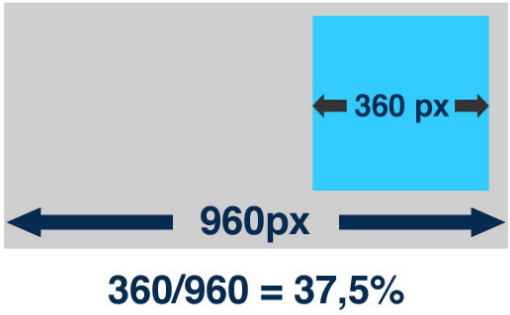
RESPONSIVE WEB DESIGN BÁSICO

En esta sección, trabajaremos en la creación de una página con diseño sensible.

El diseño sensible (según Marcotte) tiene tres componentes básicos:

- Un diseño fluido

El principal concepto en el que se apoya el **Diseño Web Adaptable** es en abandonar los anchos fijos de nuestra web. Estos deberán ser fluidos. En lugar de diseñar nuestra web basándonos en valores fijos (por ejemplo width: 960px), el diseño fluido está pensado en términos de proporciones. De esta manera cuando veamos nuestra web a través de la pequeña pantalla de un dispositivo móvil todos los elementos de la web se harán más pequeños guardando la proporción entre ellos. Por ejemplo, para saber ahora el ancho de un elemento tendremos que dividir el ancho inicial del mismo entre el ancho del elemento "padre", por llamarlo de alguna manera sencilla. Pongamos que tenemos por ejemplo esta estructura:



En este ejemplo partíamos de unos valores fijos: un contenedor de 960 pixels y dentro del mismo un elemento de 360 pixels de ancho. Si dividimos el segundo entre el primero y multiplicamos el resultado por 100 obtendremos el valor de 37,5%, que será el ancho que aplicaremos a dicho elemento. Es decir, el ancho del elemento interior será siempre el 37,5% del ancho del primero. De esta forma cuando el ancho del elemento "padre" se adapta, todos los anchos de los elementos interiores varían en función de su porcentaje. Ahora el elemento interno, en la hoja de estilos, tendrá en lugar de un width="360px" un width="37,5%".

Lo mismo haremos con los tamaños de las fuentes (por ejemplo, si el tamaño general es del 100%, que equivale a 16px, y tenemos un título de 22px, su nuevo tamaño será de 22/16 = 1.375em). ¿Pero, qué pasa con las imágenes u otros elementos que tienen un ancho fijo? Podemos adaptar su ancho así:

```
img, video, object {max-width:100%;}
```

De esta manera su ancho nunca excederá el ancho del elemento que la contiene. Y si dicho elemento cambia de ancho, también lo hará la imagen en todos los navegadores modernos. ¿He dicho modernos? Efectivamente, IE7 e IE6 no lo soportan. Para estos navegadores lo mejor es incluir en su hoja de estilos específica:

```
img, video, object {width:100%;}
```

Esta regla es completamente distinta de la anterior: Ahora decimos que la imagen (por ejemplo) siempre tendrá el mismo ancho de su contenedor. Es por ello por lo que hay que tener cuidado sobre qué elemento se aplica.

Esto está muy bien hasta que nos encontramos con anchos de pantalla realmente pequeños (por ejemplo un móvil). Si tenemos una web con tres columnas, montones de botones, menú horizontal a la derecha del logo, etc.. al comprimir tanto el tamaño de la pantalla, por mucho que los anchos sean fluidos, puede acabar todo en un caos. Es probable que tengamos que prescindir de ciertos elementos de la web o situarlos en un lugar diferente. Para ello utilizaremos los Media Queries.

MALLA FLEXIBLE

Vamos a ver cómo funciona esta fórmula dentro de un diseño de dos columnas. A continuación tenemos un elemento padre con la clase "container" que tiene dentro tanto a <section> como <aside>. El objetivo es situar <section> la izquierda y <aside> a la derecha, con márgenes iguales entre los dos. Normalmente, el estilos de este diseño se vería un poco como el siguiente.

HTML

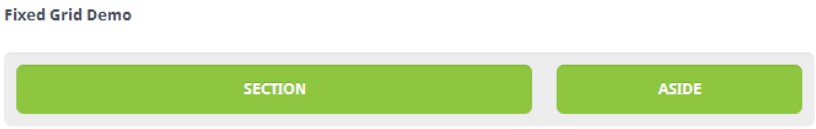
```html
<div class="container">
  <section>...</section>
  <aside>...</aside>
</div>
```

FIXED GRID

CSS

```css
.container {
  width: 660px;
}
section {
  float: left;
  margin: 10px;
  width: 420px;
}
aside {
  float: right;
  margin: 10px;
  width: 200px;
}
```

FIXED GRID



Utilizando la fórmula de la malla flexible podemos tomar todas las unidades fijas de longitud y convertirlas en unidades relativas. En este ejemplo usaremos porcentajes pero las unidades em trabajan igual de bien. Aviso, no importa cómo cambie la anchura del contenedor padre, los márgenes y anchos de <section> y <aside> escalan proporcionalmente.

```
.container {
  max-width: 660px;
}
section {
  float: left;
  margin: 1.51515151%;   /*  10px ÷ 660px = .01515151 */
  width: 63.63636363%;   /* 420px ÷ 660px = .63636363 */
}
aside {
  float: right;
  margin: 1.51515151%;   /*  10px ÷ 660px = .01515151 */
  width: 30.30303030%;   /* 200px ÷ 660px = .30303030 */
}
```

FLEXIBLE GRID

- Imágenes flexibles

Cuando las escalas de diseño hacia abajo, las imágenes y demás medios de comunicación integrados necesitan para escalar con ella, de lo contrario, podrían salir de la vista. Nos aseguraremos de que las imágenes Jenware escalar para adaptarse.

- CSS Media Queries

Como decíamos ningún diseño escala de manera adecuada cuando cambia el contexto para el que fue pensado. Los Media Queries forman parte de CSS3 e inspeccionan las características físicas del medio que va a mostrar nuestro diseño (no olvidemos que "query" equivale a "pregunta", es como preguntar: ¿qué medio se está usando?). Si las características del medio utilizado por el usuario están dentro de un condicional establecido con los Media Queries, se aplicarán una serie de instrucciones CSS contenidas dentro del mismo, de esta manera cuando nuestro diseño fluido cambia de tamaño podremos aplicar una serie de instrucciones CSS pensadas en exclusiva para ese nuevo tamaño. Vamos a ver un ejemplo.

El ancho de pantalla actual del iPhone es de 320px. Vamos a suponer que para ese ancho nuestro diseño fluido presenta una serie de dificultades (puede ser desde cambiar el logo, eliminar una columna, cambiar la organización de los elementos de la pantalla, etc…). Dentro de nuestra hoja de estilos haríamos:

```
@media screen and (max-width: 320px)
{
/* Aquí van las reglas CSS necesarias */
}
```

La instrucción se compone de dos partes: el tipo de medio utilizado (o Media Type, en este caso "screen", que agrupa a todos los medios que se ven vía una pantalla) combinándolo mediante un "and" con el Media Query (max-width: 320px). Estamos preguntando: ¿Es un medio con pantalla y tiene un ancho de 320px o menor? Entonces le aplicamos los estilos situados entre los corchetes correspondientes.

Podemos empezar desde este ancho e ir subiendo a otras posibles opciones. Algunos autores recomiendan optimizar estos anchos de pantalla:

- 320 px
- 480 px
- 600 px
- 768 px
- 900 px
- 1200 px

Características de los medios que se pueden evaluar con Media Queries

Característica	Descripción
width	Ancho de la zona de visualización (viewport o ventana gráfica).
height	Alto de la zona de visualización (viewport o ventana gráfica).
device-width	Ancho de la superficie de representación dispositivos (toda la pantalla).
device-height	Ato de la superficie de representación dispositivos (toda la pantalla)
orientation	Si el dispositivo está en orientación vertical (portrait) u horizontal (landscape). (No acepta prefijos min-/max-.)
aspect-ratio	Relación entre la anchura de la ventana (viewport)dividida por la altura (ancho / alto).
device-aspect-ratio	Relación entre el ancho de la pantalla completa del (superficie de representación), y la altura.
color	La profundidad de bits de la pantalla
color-index	El número de colores de la tabla de colores de consulta.
monochrome	El número de bits por pixel en un dispositivo monocromo.
resolution	La densidad de píxeles en el dispositivo. Esto es cada vez más relevante para la detección de pantallas de alta resolución
scan	Si un dispositivo de tv soporta barrido progresivo o entrelazado. (No acepta prefijos min-/max-.)
grid	Si el dispositivo utiliza una pantalla basada en grid o malla, tal como una fuente de ancho fijo. (No acepta min-/maxprefixes.)

Navegadores que soportan Media Query

En general todos los navegadores modernos lo soportan. Eso quiere decir que Internet Explorer 6, 7 y 8 no lo soportan (Tan raro en IE!). Afortunadamente hay soluciones utilizando Javascript, por ejemplo respond.js. **respond.js** proporciona un script rápido y ligero (3kb minified / 1kb gzip) que permite utilizar diseños web adaptables en navegadores que no soportan CSS3 Media Queries – en concreto Internet Explorer 8 e inferiores.

INICIAMOS…

El objetivo de la práctica es crear una plantilla en HTML5 que cumpla con los criterios de diseño sensible.

Para comenzar, crearemos la página y sus estilos de manera tradicional, especificando valores estáticos, para luego ir añadiendo los elementos de diseño sensible.

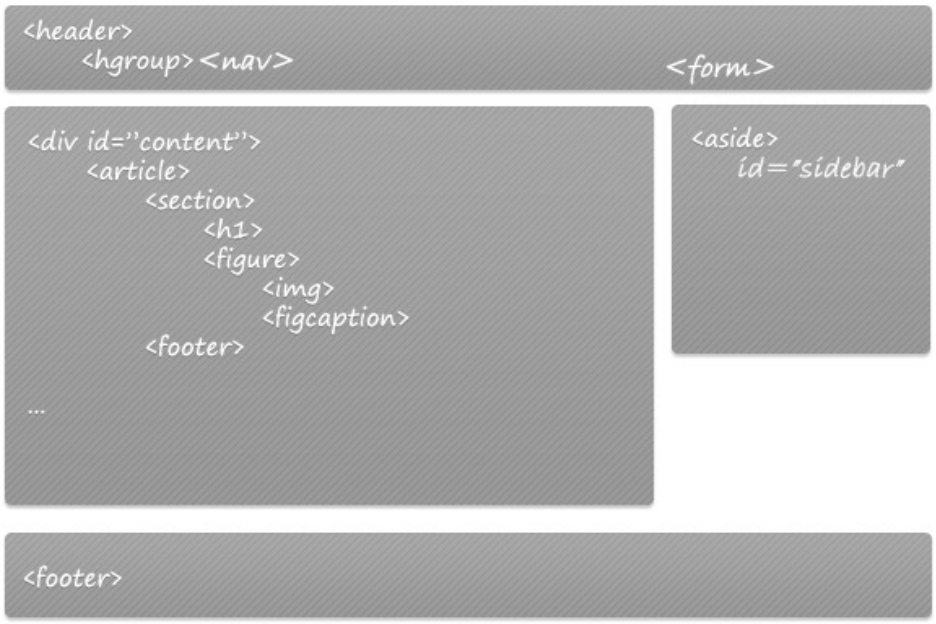
A esta plantilla se le dotará de los siguientes elementos de estilo:

- Logo simple usando Google Font API[1][2]
- Efectos de sombra.
- Gradientes CSS3. [3]

[1] http://www.red-team-design.com/google-font-api-and-typekit-solutions-vs-font-face
[2] http://www.google.com/fonts/ https://developers.google.com/fonts/docs/getting_started

- Patrón de fondo utilizando el esquema de datos URI (data URI scheme)[4][5].
- Flecha hechas con pseudo-elementos.

La plantilla tendrá la siguiente estructura:



1) Creando la cabecera – Doctype y Meta

Comencemos por definir el tipo de documento a HTML5 y la creación de la sección head con las secuencias de comandos y los archivos CSS. La declaración de tipo de HTML5 es bastante fácil de recordar en comparación con versiones en HTML / XHTML anteriores.

```
<!DOCTYPE HTML>
```

En nuestra sección head vamos a configurar el juego de caracteres UTF-8, que también es más fácil ahora en HTML5

```
<meta charset="UTF-8">
```

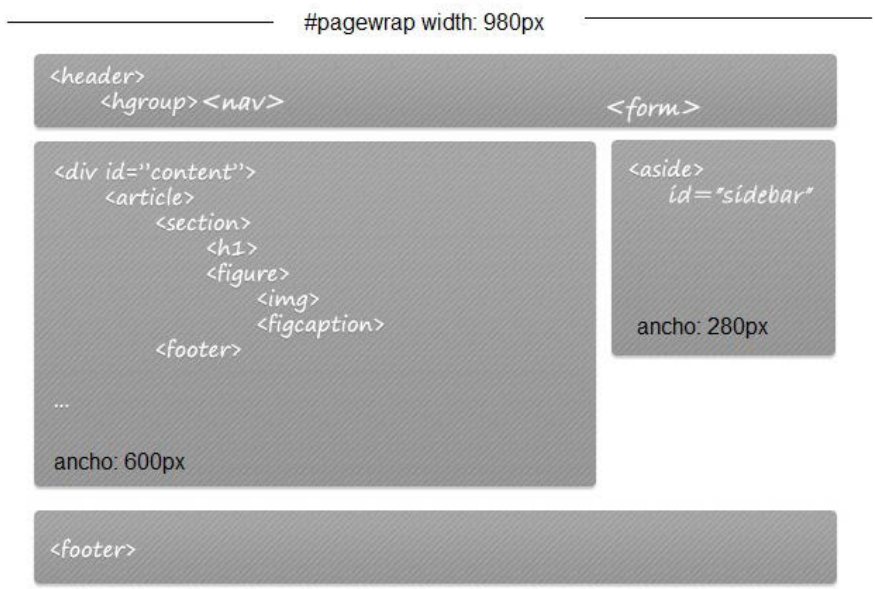
2) Creando la estructura del cuerpo

El cuerpo de la plantilla se compone de dos bloques:

<aside> - que envuelve a la navegación

 - que contiene los elementos <article>.

[3] Puede utilizar herramientas online como **Ultimate CSS Gradient Generator.**
http://www.colorzilla.com/gradient-editor/
[4] http://es.wikipedia.org/wiki/Data:_URL
[5] Obtener esquema de datos URI desde imagen.
http://websemantics.co.uk/online_tools/image_to_data_uri_convertor/



Tenga en cuenta que ahora que tenemos estos nuevos elementos HTML, no hay que olvidar por completo el elemento ya que aún tiene funciones para realizar.

```
<div id="pagewrap">

	<header id="header">

		<hgroup>
			<h1 id="site-logo"><a href="#">Pr&aacute;tica 3</a></h1>
			<h2 id="site-description">Crear una plantilla HTML con dise&ntilde;o sensible</h2>
		</hgroup>

		<nav>
			<ul id="main-nav" class="clearfix">
				<li><a href="#">Home</a></li>
			</ul>
		</nav>

		<form id="searchform">
			<input type="search" id="s" placeholder="Search">
		</form>

	</header>

	<div id="content">

		<article class="post clearfix">
    <header>
				<h1 class="post-title"><a href="#">Objetivo de la práctica</a></h1>
				<p class="post-meta"><time class="post-date" datetime="2011-05-08"
pubdate>Abril 8, 2013</time> <em>in</em> <a href="#">Categoría</a></p>
			</header>
			<figure class="post-image">
				<img src="images/ejemplo.jpg" />
			</figure>
			El objetivo de la práctica es crear una plantilla en HTML5 que cumpla con los
criterios de diseño sensible. A la vez a esta plantilla se le dotará de elementos de estilo.
		</article>

	</div>

	<aside id="sidebar">

		<section class="widget">
```

```
                        <h4 class="widgettitle">Sidebar</h4>
                        <ul>
                                <li><a href="#">WordPress</a> (3)</li>
                                <li><a href="#">Design</a> (23)</li>
                                <li><a href="#">Web </a>(18)</li>
                        </ul>
                </section>

        </aside>
</div>
```

Añadir pie de página principal

```
<footer id="footer">
                Tomado de <a href=" http://webdesignerwall.com/tutorials/responsive-design-with-css3-
media-queries " target="_blank" > WebDesignerWall </a>
        </footer>
```

3) AÑADIR EL CSS

Restablecer los elementos HTML5 a Block.

El siguiente código CSS restablecerá los elementos de HTML5 (article, aside, figure, header, footer, etc.) a block.

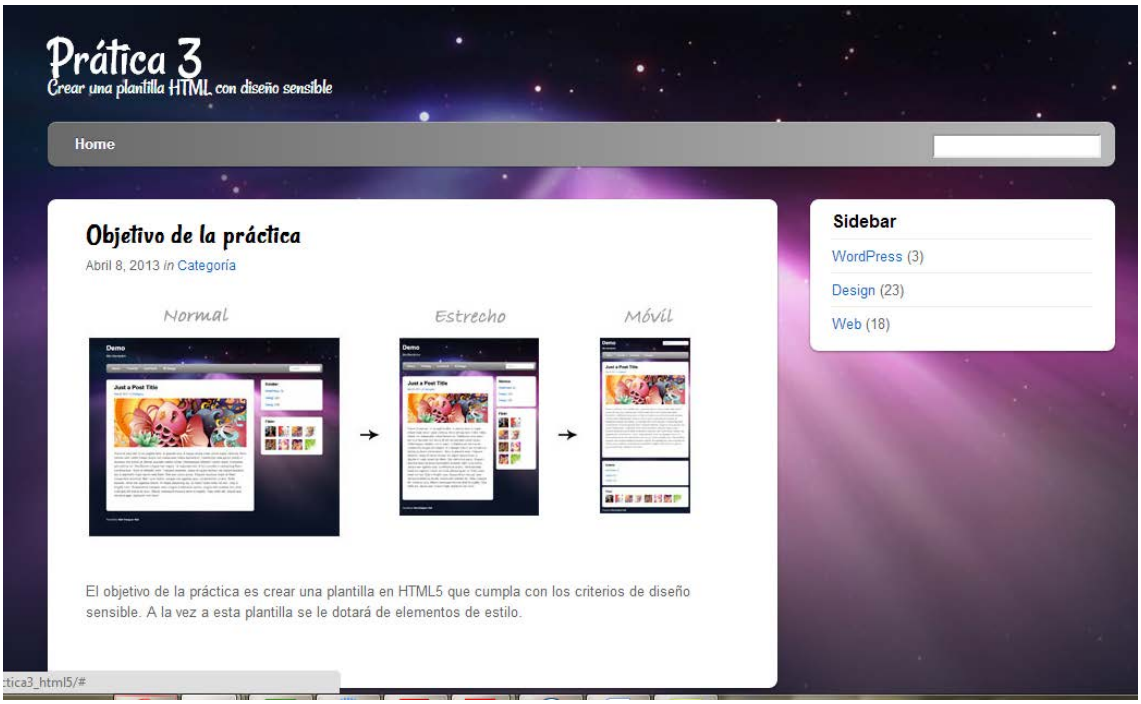
```
article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section {
   display: block;
}
```

Principal contenido CSS

El contenedor principal "pagewrap" tiene 980px de ancho. <header> tiene 160px de altura fija. El contenedor "contenido" es 600px de ancho. El contenido de "sidebar" es 280px de ancho.

```
#pagewrap {
        width: 980px;
        margin: 0 auto;
}

#header {
        height: 160px;
}

#content {
        width: 600px;
        float: left;
}

#sidebar {
        width: 280px;
        float: right;
}

#footer {
        clear: both;
}
```

El resultado sería algo así:



4) Agregar compatibilidad de HTML5 a IE

Una vez que se han incluido los archivos CSS, se deben incluir los scripts necesarios. Como estamos utilizando características de HTML5 y CSS3 necesitamos algunos scripts para que estas características puedan verse en todos los navegadores. El primer script que se va a utilizar es **Modernizr.js**[6], una biblioteca de detección de características de HTML5 y CSS3. Modernizr.js permite fácilmente detectar si el navegador soporta una función específica o no. Entra aquí y selecciona las características que quiere habilitar y luego descargar.

El segundo script que necesitamos es **Respond.js,** un script que permite utilizar responsive design (diseño sensible) en Internet Explorer y otros navegadores que no soportan CSS Media Queries.

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<link rel="stylesheet" href="style.css" type="text/css" media="screen" />
<link href='http://fonts.googleapis.com/css?family=Open+Sans|Baumans' rel='stylesheet' type='text/css'/>

<script src="js/modernizr.js"></script>
<script src="js/respond.min.js"></script>
```

Hasta este punto, tenemos una plantilla HTML5 normal. Ahora vamos a convertirla en una que sea adaptable o sensible.

Una página con diseño sensible se comporta como se muestra en la imagen siguiente, que vemos que la página se adapta a los anchos de pantalla de los diferentes dispositivos.

[6] http://modernizr.com/download/

5) Establecer el *viewport*

Como queremos crear un diseño sensible que funcione en todo tipo de dispositivos y resoluciones de pantalla, tenemos que añadir la etiqueta meta **viewport** que define cómo el sitio web se debe mostrar en un dispositivo. Establecemos el ancho de device-width y la escala inicial de 1,0. Lo que esto hace es ajustar la anchura de la ventana a la anchura del dispositivo y ajustar el nivel de zoom inicial a 1,0. De esta forma el contenido de la página se mostrará 1:1, una imagen con un tamaño de 350 píxeles en una pantalla con un ancho de 350 píxeles llenará todo el ancho de la pantalla.

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Tenga en cuenta que hay varias opiniones diferentes sobre el uso de initial-scale y width = device-width. Algunos dicen no usar initial-scale en absoluto, ya que podría conducir a un comportamiento incorrecto en iOS. En algunos casos se produce un error de zoom cuando se gira el dispositivo de posición vertical a horizontal. Como resultado los usuarios tienen que quitar el zoom manualmente. Quitar la propiedad initial-scale a veces puede corregir ese error. Si no funciona, hay una secuencia de comandos que desactiva la capacidad del usuario para escalar la página, permitiendo que el cambio de orientación se produzca correctamente.

- initial-scale=1.0 ajusta el viewport (ventana gráfica) a las dimensiones del dispositivo (valores de altura-dispositivo y ancho-dispositivo), lo que es una buena idea porque el tamaño de la ventana se ajusta a las dimensiones del dispositivo, independientemente de su orientación.
- width=device-width el viewport (ventana gráfica) para que corresponda siempre al ancho (valor fijo) del dispositivo, por lo que se distorsiona en orientación horizontal (landscape),porque el valor correcto debería ser la "altura del dispositivo " y no el "ancho del dispositivo " en la orientación horizontal (y es peor en iPhone5 donde el valor de la altura del dispositivo es 568px, comparado con su ancho 320px)

6) Media Queries[7]

El truco para diseño sensible son los CSS3 Media Queries. Es como escribir condiciones "if" diciéndole al navegador cómo mostrar la página para el ancho de ventana especificado.

Crear una nueva hoja de estilo para las media queries.

```
<link href="css/media-queries.css" rel="stylesheet" type="text/css">
```

Viewport < 980px (diseño fluido)

Un Viewport más estrecho que 980px, se le aplicarán las siguientes reglas:

[7] Ver cómo funcionan las media queries http://webdesignerwall.com/tutorials/css3-media-queries

- pagewrap = restablezca ancho a 95%
- content = restablezca ancho a 60%
- sidebar = restablezca ancho a 30%

Consejos: Usar valores de porcentaje (%) para hacer los diseños fluidos.

```
@media screen and (max-width: 980px) {

        #pagewrap {
                width: 95%;
        }

        #content {
                width: 60%;
                padding: 3% 4%;
        }

        #sidebar {
                width: 30%;
        }
        #sidebar .widget {
                padding: 8% 7%;
                margin-bottom: 10px;
        }

}
```

980px -> 95%
600px -> ¿?
(600*95)/980=58,16 ~60%

VIEWPORT < 650PX (DISEÑO DE UNA COLUMNA)

header = restablece altura a auto

searchform = reposicionar searchform a 5px top

main-nav = restablecer la posición a static

site-logo = restablecer la posición a static

sitio-description = restablecer la posición a static

content = restablece el ancho en automático (esto hará que el contenedor se expanda a su ancho total) y deshacerse del float

sidebar = restablecer ancho al 100% y deshacerse del float

```
@media screen and (max-width: 650px) {

        #header {
                height: auto;
        }

        #searchform {
                position: absolute;
                top: 5px;
                right: 0;
        }

        #main-nav {
                position: static;
        }

        #site-logo {
                margin: 15px 100px 5px 0;
                position: static;
        }

        #site-description {
```

```
                margin: 0 0 15px;
                position: static;
        }

        #content {
                width: auto;
                float: none;
                margin: 20px 0;
        }

        #sidebar {
                width: 100%;
                float: none;
                margin: 0;
        }

}
```

VIEWPORT < 480PX

El siguiente CSS se aplicará si la ventana es más estrecha que 480px, que es el ancho de la pantalla del iPhone en posición horizontal.

html = desactivar el ajuste del tamaño del texto. De forma predeterminada, iPhone agranda el tamaño del texto para que se lea con más comodidad. Puede desactivar el ajuste de tamaño de texto, añadiendo-webkit-text-size-adjust: none;

main-nav = restablecer el tamaño de la fuente al 90%

```
@media screen and (max-width: 480px) {

        html {
                -webkit-text-size-adjust: none;
        }

        #main-nav a {
                font-size: 90%;
                padding: 10px 8px;
        }

}
```

HACER FLEXIBLES LAS IMÁGENES

Para que las imágenes sean flexibles, sólo se tiene que añadir max-width: 100% y height: auto. Los valores max-width: 100% y height: auto funcionan en IE7, pero no en IE8 (sí, otro error IE raro). Para solucionar este problema, es necesario agregar width: auto \ 9 de IE8.

```
img {
        max-width: 100%;
        height: auto;
        width: auto\9; /* ie8 */
}
```

VIDEOS INTEGRADOS FLEXIBLES

Para hacer que los vídeos incrustados sean flexibles, se utiliza el mismo truco antes mencionado. Por razones desconocidas, max-width: 100% (para el elemento embed) no funciona en Safari. La solución es utilizar width: 100% en su lugar.

```
.video embed,
.video object,
.video iframe {
        width: 100%;
        height: auto;
}
```

Internet Explorer 8 o versiones anteriores no son compatible con Media Queries. Usted puede utilizar respond.js o media-queries.js para añadir compatibilidad con Media Queries en IE.

```
<!--[if lt IE 9]>
        <script src="http://css3-mediaqueries-js.googlecode.com/svn/trunk/css3-mediaqueries.js"></script>
<![endif]-->
```

COMO PASO FINAL:

- Buscar un hosting o alojamiento gratis.
- Subir los archivos html y css que resultaron de la práctica.
- Ir al sitio http://quirktools.com/screenfly/ , copiar el enlace de su sitio probar su página en los diferentes dispositivos disponibles

LECTURAS RECOMENDADAS

Responsive Web Design

Don't do this in responsive web development

Build a basic responsive site with CSS

Create a stylish HTML5 template from scratch

Create a Responsive Web Design Template

Building fast and responsive websites

Cómo funcionan las media queries

Diseño web sensible, responsive web design o diseño adaptativo

Responsive Design in 3 Steps

15 TUTORIALES PARA APRENDER A DISEÑAR UN SITIO WEB RESPONSIVE.