JUUNTOS.ORG

Responsive Web Design wtf!

La proliferación y el protagonismo que desde hace algunos años tienen los dispositivos móviles en nuestra vida cotidiana, sumado a la expansión de la WWW, obligan a los desarrolladores a plantearse nuevas formas de diseño. En los últimos tiempos han aparecido técnicas dentro de las cuales el "diseño responsivo" resalta por sobre las demás constituyendo una realidad palpable y visible en incontables páginas webs. Este dossier pretende dar un acercamiento sobre su basamento, sus características, implicancias y significados.

Agradecimientos:

- A *Ethan Marcotte* por su gran contribución al desarrollo web
- A *John Allshopp* por sus textos inspiradores
- A *James Burke* en cuyos documentales me inspiro y me apoyo para hacer los míos propios
- A toda la comunidad Juuntos por el empuje y la amistad

¡wTf!

Antes de adentrarnos en el tema que convoca este dossier, vamos a explicar concisamente de qué estamos hablando.

La palabra castellana *responsivo* es definida por la **RAE**[1] como "*perteneciente o relativo a la respuesta*". Si bien en ésta hay un acercamiento hacia su significado –por ello la utilizo, más allá de lo que digan[2]-, es más sencillo entenderlo traduciendo el término inglés "*responsive*" cuyas acepciones son: "*receptivo, sensible*".

Con estos primeros esbozos daremos una definición ampliada y mas precisa, cual sería: ***"conjunto de técnicas de diseño y programación que facilitan a una interfaz adaptarse a diversas resoluciones de pantalla según sea el dispositivo que la contenga".***

Esto quiere decir que una web creada mediante la técnica *responsiva* se verá igual o muy similar, se adaptará según sea el dispositivo desde donde se la llame, tendrá las mismas funcionalidades, la misma accesibilidad y usabilidad; será "*sensible*" a su entorno, dará la misma "*respuesta*".

¿Utopía? No, ya existen.

Ejemplos experimentables:

Utilizando cualquier tipo de computadora o dispositivo y un navegador web, accedan a alguno/s de los enlaces indicados más abajo; si es posible, varíen el ancho de la pantalla, allí verán la acción:

Webs Joomla (plantillas):

- Joomla Art: http://joomla25-templates.joomlart.com/ja_elastica/
- Shape5: http://www.shape5.com/demo/index.php?vertex

Webs Wordpress (plantillas):

- Base: http://wordpress.org/extend/themes/responsive
- http://jinsonathemes.com/fabs/?themedemo=Vasiliki
- http://www.tripwiremagazine.com/2012/07/wordpress-mobile-theme.html

Webs Drupal:

- Omega: http://omega.developmentgeeks.com/
- Respondr: http://www.sooperthemes.com/drupal-themes/respondr
- Adaptative: http://adaptivethemes.com/

Otras:

- 60 Ejemplos: http://inspirationfeed.com/inspiration/websites-inspiration/60-examples-of-responsive-website-design/

¿De dónde salió esto?

Antecedentes

En los albores de internet, una página web era una representación digital de un papel impreso, una imitación de aquello que estaba escrito sea en formato de libro, lámina o dibujo.

Con el paso del tiempo y por necesidades imperantes de compartir contenidos más complejos, fueron mejorando no solo los lenguajes para su creación sino que además, y por consecuencia directa, los dispositivos para verlas y el software para interpretarlas.

Para evitar la dispersión, mantener un orden, y seguir normas y criterios de desarrollo, surgió en 1994 la **W3C**[3] (World Wide Web Consortium); una comunidad internacional que produce recomendaciones acerca de internet, siendo sus principios: *web para todo el mundo, web en cualquier dispositivo; y su visión: web de autores y consumidores, datos y servicios, confianza*.

La reciente historia de la tecnología nos dice que hasta no hace mucho tiempo solo podíamos ver páginas web en monitores o pantallas para escritorio. Esto podía significar para el desarrollador tener solo un pequeño puñado de obstáculos salvables más o menos sin mayores complicaciones.

A partir de la aparición de nuevos navegadores o browsers, del auge de los dispositivos móviles y de la gran expansión de la red de redes y su acceso masivo, los problemas fueron creciendo casi a la par, cuando no exponencialmente.

Por lo tanto, la complejidad en la creación y desarrollo web tomó un rumbo que muchas veces escapó y escapa a cualquier control que se quiera imponer, y deja la imaginación como único medio para librarnos de escollos cada vez más crecientes.

Como risible argumentación ampliatoria, y tomando en cuenta que una página o portal web constituye un "*sistema*", podemos acotar la siguiente Ley de Murphy[4]: "*los sistemas son inherentemente imposibles de probar. Es posible probar completamente sólo sistemas muy elementales. Para el resto, las pruebas exploraran sólo una ínfima fracción de las situaciones posibles*".

Para suplir algunas de las falencias que se presentan, surgieron diferentes técnicas de diseño y programación, muchas de las cuales aún están vigentes. Quizás lo que más se conoce por su gran uso y difusión sean las maquetaciones CSS múltiples. Esta técnica consiste en asignar a la página o portal web, programación mediante, un CSS creado ad hoc según sea el dispositivo detectado o según sea el browser o navegador que el usuario esté utilizando.

Con el mismo fin, existe toda una parafernalia de "*hacks CSS*"[5] que nos permiten hacer pequeñas correcciones para que la presentación de la página web, su funcionalidad y su estilo, sea igual o similar en la mayoría de los navegadores.

Otro uso extendido lo constituyen los *frameworks*[6] CSS utilizados para facilitar el trabajo del diseñador y programador, como por ejemplo Blueprint[7].

Así y todo no alcanzamos a tapar los agujeros que existen ni a cumplir en su totalidad las expectativas de los usuarios, muchos menos a lograr sortear todos los obstáculos que nos imponen las nuevas tecnologías en auge, aunque las tendencias y experimentaciones van apuntando para llegar a esa meta.

Actualidad

El concepto de "*Responsive Web Design*" fue utilizado públicamente la primera vez en el 2010 por *Ethan Marcotte*[8], quien redactó un artículo[9] homónimo para la revista especializada "*A List Apart*". Más tarde publica un libro[10] con el mismo título explayándose en las bases inspiradoras que le dan origen y con un contenido más técnico, práctico y palpable.

Ethan parte de otro texto[11] escrito tiempo atrás, en abril del 2000, por *John Allshopp*, quien conjuga de manera interesante el Tao Te Ching con el diseño web. Suma, como iluminación y visión, la nueva disciplina arquitectónica llamada "*arquitectura responsiva*"[12].

(Vale destacar que el texto de *John Allshopp* es atrapante por demás; hondamente interesante, vivificante y motivador; nos da otro enfoque distinto en el camino al mejor diseño. Y aunque han pasado los años, lo que allí dice sigue plenamente vigente, deberían leerlo).

Lo expuesto por *Ethan* es el resultado de estudios y análisis provenientes de diversas fuentes, incluso de otros textos escritos con anterioridad por él mismo y por otros autores, que apuntan a temas relacionados como por ejemplo "*Fluids grids*"[13] o "*Mobile First*"[14] de *Luke Wroblewski*, ambos del 2009.

Luego de estas primeras luces recientes del RWD -que nos abrieron la mente-, son incontables los autores, blogs o webs, en donde se aborda el tema desde diferentes puntos de vista, con algunas buenas aportaciones y con discusiones al respecto, porque en estos procesos evolutivos todo es vértigo.

Conclusiones

Los textos escritos y comentados, constituyen en definitiva planteamientos y desafíos, soluciones que aparecen como mágicas, que despiertan polémica o nos alegran, pero que persiguen un fin único: la real pero aún no concluida tarea de independizar una página web de cualquier hardware/software desde donde se la llame, facilitando nuestro cotidiano devenir, dándonos la celebérrima y renombrada mejor ***experiencia de usuario***.

Como todo lo nuevo que aparece, tiene sus precursores y detractores, y aunque la pulseada va a favor del *RWD*, el futuro dirá en qué decantará esta nueva tecnología, en dónde estará nuestro próximo horizonte que apunta a lo infinito.

Veamos de qué se trata en detalle puesto que como decía James Burke (ver foto debajo), todo lo que hay de nuevo surge de "***conexiones***"[15].

La base estaba

El concepto de *RWD* parte de tres reglas o principios fundamentales propuestos y prexistentes cuyos orígenes provienen de diversos autores, y son:

- ***diseñar para móviles en primer lugar***,
- ***mejora progresiva***
- ***paradigma javascript no obstructivo***.

Diseñar primero para móviles

El que describe las razones de diseñar primero para móviles es *Luke Wroblewski*[16], quien llega a definir este concepto como *fundamental* partiendo de conclusiones tomadas en base a estadísticas sobre dispositivos móviles y de otras evidencias actuales cuales son:

- Los móviles están creciendo de manera exponencial.
- Los móviles obligan a los desarrolladores a concentrarse solo en los datos y las acciones puesto que las reducidas pantallas no dan para otra cosa, es una cuestión de prioridad.
- Las aplicaciones webs interactivas para móviles son cada día más sorprendentes y han ampliado largamente el limitado horizonte original de la web tal como la conocíamos.
- Algunas plataformas móviles y sus navegadores sobrepasan las capacidades y dejan obsoletos a los de escritorio o desktop.

Mejora progresiva

El término fue acuñado en el año 2003 por *Steven Champeon*[17] de la firma de diseño web *Hesketh*.

La mejora progresiva hace referencia a una *"estrategia particular de diseño web que acentúa la accesibilidad, margen de beneficio semántico, y tecnologías externas del estilo y el scripting, en una manera adecuada que permite que cada uno tenga acceso al contenido y a la funcionalidad básica de una página web, usando cualquier navegador web o conexión a Internet, mientras que también permite a otros con un mayor ancho de banda o un navegador web más avanzado experimentar una versión mejorada de la página"* (*Wikipedia*)[18].

Es la inversa a la "*degradación agraciada*" que plantea diseñar primero para los navegadores más recientes e ir agregando técnicas y códigos para que la web también funcione en otros más obsoletos o con limitaciones tecnológicas; técnica utilizada sobradamente en estos tiempos.

Al contrario, una web con "*mejora progresiva*" significa que partirá de una base funcional y estética visible y presentable en cualquier navegador y se irá haciendo más completa y mejor según sea el dispositivo donde se muestre, logrando esto último mediante el uso de tecnologías modernas como Java Applets, Flash, CSS, SVG (imágenes escalables), etc.

La "*mejora progresiva*" supone respetar una serie de reglas o principios básicos cuales son:

- Todo el contenido básico debe ser accesible a todos los browsers.
- Toda la funcionalidad básica debe ser accesible a todos los browsers.
- Ser discreto, el marcado semántico[19] debe estar en todos los contenido.
- Las plantillas o vistas mejoradas deben ser provistas por CSS externos enlazados.
- El comportamiento mejorado debe ser no obstructivo, con javascript enlazado externamente.
- Las preferencias del navegador del usuario final deben ser respetadas.

Javascript no obstructivo

Es un paradigma floreciente en el uso del lenguaje de programación javascript utilizado en la web. Aunque el término no está definido formalmente en ningún lado, sus principios generalmente incluyen:

- Separación de la funcionalidad JavaScript (la "capa del comportamiento") de las capas de estructura/contenido y de presentación de un página.

- Uso de buenas prácticas a fin de evitar los problemas de incompatibilidad de la programación tradicional en JavaScript (tales como inconsistencias entre navegadores y falta de escalabilidad).

Conexión

Al final del capítulo anterior dije que todo se daba por conexiones, justamente las técnicas mencionadas de programación y diseño web están presentes entre nosotros desde hace algún tiempo, no son para nada inventos salidos de la nada sino sucesivos avances crecientes a partir de lo anterior. Su uso está supeditado y es aplicado según cada desarrollador, tal como sean sus alcances y conocimientos o "*skills*"[13].

Por lo tanto, podemos decir que llegar a dominar el diseño web actual respetando estas tres reglas o bases supone poseer algunas "*expertises*"[20] en HTML, CSS y javascript; además de aquellas relacionadas al entorno de operatividad –llámese framework- en donde trabaja, como pueden ser *Joomla*, *Drupal*, *Wordpress*, etc. y sus lenguajes de programación relacionados.

RWD y Diseño fluido

A *prima facie* un experto podría decir que RWD es lo mismo que crear una maquetación CSS fluida o autoajustable pero lamento decepcionarlos, no es lo mismo, veamos un poco el tema.

Hasta el momento hay cuatro formas de crear una web:

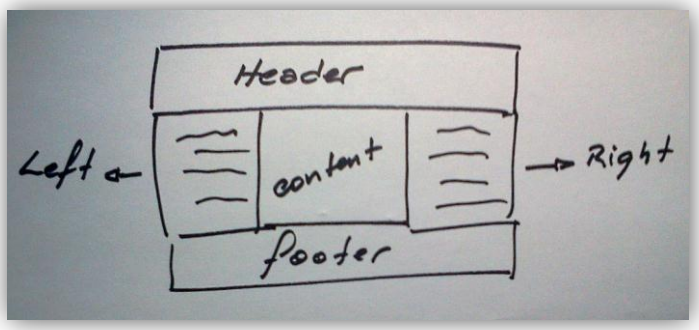
- mediante maquetación fija (ancho fijo),
- maquetación variable o fluida (ancho variable),
- una mezcla de ambas,
- el flamante RWD.

Maquetación rígida o fija

Cuando trabajamos con maquetación fija, lo que estamos haciendo es darle a los distintos espacios o sectores que conformarán nuestra web un ancho "fijo" basado en pixeles de modo que definimos, por ejemplo:

- Ancho total de la web: 960px
- Cabecera: 960px
- Barra o lateral izquierdo: 250px
- Barra o lateral derecho: 250px
- Centro para contenidos: 460px
- Pie: 960px

El "*mock-up*"[21] sería más o menos este:



De esta forma la web se verá perfectamente bien a partir de monitores con una resolución de 1024px x 768px –actualmente los más comunes- con lo cual nos garantizamos que en pantallas con resoluciones superiores no perderá la forma e igualmente se verá bastante bien.

Problemas de la maquetación fija o rígida

- Puede que la página se vea muy chica si el monitor o pantalla tiene una alta resolución, dejará muchos espacios en blanco o vacíos.
- Viceversa. Puede que la página sea muy grande si la pantalla tiene una resolución menor por lo que podría resultar engorrosa o resultar con una mala estética.
- En navegadores de dispositivos móviles tal vez se vea bien, pero seguramente será tan pequeño que el usuario necesitará una lupa para leer.

Maquetación variable, autoajustable o fluida

En estos casos nuestra maquetación será basada en porcentajes y bien podemos tener por ejemplo:

- Ancho total de la web: 100%
- Cabecera: 100%
- Barra o lateral izquierdo: 20%

- Barra o lateral derecho: 20%
- Centro para contenidos: 60%
- Pie: 100%

Haciendo nuestro trabajo de este modo nos aseguramos que sin importar el ancho de la pantalla que utilice el usuario los contenidos de cada sección se verán bien ya que se autoajustarán automáticamente.

Problemas de la maquetación fluida

- Si el contenido es escaso perderá la estética.
- Si la pantalla a utilizar es reducida y los contenidos son largos, aparecerán las temidas barras de desplazamiento, odiadas por todos.
- En dispositivos móviles, si bien nuestra web se auto ajustará, una imagen grande o el tamaño de la tipografía pueden hacer que se descompagine, se vea mal, sea poco funcional, no accesible, no usable e incómoda.

Maquetación mixta

Hay muchos casos en donde se combinan ambas técnicas definiendo espacios fijos y otros variables, los unos en pixeles, los otros en porcentajes. Así encontramos webs cuyo lateral izquierdo, por ejemplo, está maquetado en pixeles y el resto en porcentajes. Esto generalmente se hace para mantener menús o secciones de publicidades con un tamaño fijo y no perder la estructuración deseada.

Los problemas que surgirán podrán ser similares a los comentados más arriba o una mezcla de ambos.

RWD

El *diseño responsivo* ofrece todo el control que tendríamos con un diseño de ancho fijo y nos ofrece muchísima más flexibilidad que el diseño fluido. Nos permite desarrollar páginas que se visualicen de forma correcta en cualquier dispositivo y resolución.

Es decir, podemos trabajar tanto en forma rígida o fija como fluida, con la diferencia que el control será nuestro siempre, garantizando no solo una correcta visión de la página web sino que además su funcionalidad.

Mientras que el diseño fluido nos da un nivel de navegabilidad y usabilidad muy bueno, el diseño responsivo nos da la excelencia ya que en vez de basarse en porcentajes se basa en proporciones.

Cómo se crea el RWD

El diseño responsivo se crea a partir de tres métodos o técnicas de programación que conforman sus pilares esenciales:

1. **Una grilla flexible,**
2. **Imágenes y medios flexibles,**
3. ***Media queries*, un módulo de las especificaciones del CSS3.**

Grilla flexible

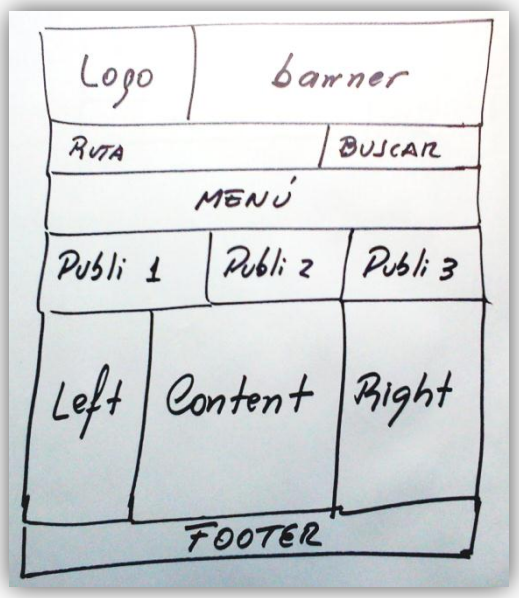
Se entiende por grilla flexible a la división de la web en sectores iguales y/o proporcionales.

He nombrado antes a los frameworks CSS pero incluso podemos crearnos una grilla a nuestro gusto y paladar, aunque ello sería motivo de un tutorial aparte por lo que no lo vamos a tratar aquí. Y no todos los frameworks CSS son responsivos, *Blueprint* por ejemplo no es de este tipo.

Los "*grids flexibles*" más conocidos son los siguientes:

- **Simplegrid**: http://simplegrid.info/
- **Less Framework 4**: http://lessframework.com/
- **Bootstrap**: http://twitter.github.com/bootstrap/
- **Foundation 3**: http://foundation.zurb.com/
- **Skeleton**: http://www.getskeleton.com/
- **InuitCSS**: http://csswizardry.com/inuitcss/
- **YUI Grids:** http://developer.yahoo.com/yui/grids/
- **960gs:** http://960.gs/
- **Gumby Framework:** http://www.gumbyframework.com/

En un sentido básico, un "*grid*" o grilla CSS está compuesto por una serie de clases CSS que definen espacios proporcionales e iguales en el diseño. Para hacer un acercamiento, imaginen una hoja cuadriculada en donde cada rectángulo, o un grupo de ellos, tendrían una clase CSS y un tamaño asignado; visualmente algo parecido a lo que se utiliza en Photoshop para "marcar" una imagen y luego cortarla con la herramienta cuchillo.



Para poder trabajar con grillas, lo primero que hay que tener bien en claro es qué elementos o espacios contendrá la web a realizar. Debemos tener nuestro *mock-up* o boceto definido y afinado. Un ejemplo esquemático pueden ver a la derecha de este texto.

Los porcentajes

Los espacios, casilleros o grupos de ellos que definamos en nuestra grilla tendrán asignados tamaños en porcentajes. Si estamos definiendo una web con ancho fijo y queremos que dichos espacios sean responsivos, haremos un cálculo matemático simple, por ejemplo:

```
Ancho de la web: 960px
Ancho del espacio para banners: 120px
Cálculo: (120 / 960) x 100 = 12,5%
(con alguna que otra variación según tengamos o no paddings o margins).
```

Las proporciones

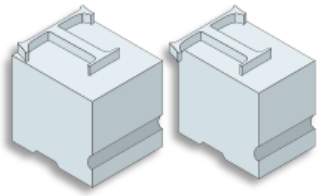
A todos los demás elementos del diseño debemos asignarles medidas en "*em*" ya que los valores así puestos son calculados en base al tamaño del elemento padre y tienen la facultad de heredar su propiedad. Si un * * tiene un tamaño de fuente de 16px cualquier elemento dentro de esa capa —hijo— heredará el mismo tamaño de fuente a menos que se la modifique específicamente.

Por ejemplo, si el tamaño de fuente del elemento hijo se cambia a 0.75em entonces el valor para el tamaño será de 0.75 * 16px = 12px. Si el usuario incrementa (o reduce) el tamaño del texto en su navegador, la interface entera se reducirá o ampliará.

De dónde lo de "em" o "pica"

La unidad de medida "*em*"[22] tuvo sus orígenes en la imprenta y equivalía al tamaño del tipo de la letra "M", algo así como 1/8 de pulgada.

Esto podría hacer pensar al experto que en los textos de las computadoras las letras no ocupan el mismo espacio, lo cual es verdad y por ello se utiliza el "*kerning*"[23], para ajustar el espacio horizontal y hacer que cada letra equidiste y sea balanceada en la pantalla.



En las imprentas, originalmente el tipo era recortado o "*kerned*" para igualar este espacio horizontal de cada letra (ver figura).

De modo que en CSS, un "*em*" es en realidad una "*medida vertical*" siendo igual al espacio vertical necesario para cualquier letra dada de una fuente, en referencia al espacio horizontal que esta ocupa: si el tamaño de letra es 16px, entonces 1em = 16px.

Las pistas para seguir...

Sabemos que los navegadores más populares tienen un tamaño de fuente que por defecto es el ya mencionado 16px cuya equivalencia es 1em (en Firefox se puede ver el tamaño de fuente por defecto accediendo al menú *Herramientas -> Opciones -> Contenido).*

Esto quiere decir que antes de escribir cualquier selector simple en CSS, el navegador ya tiene un tamaño de fuente de 16px por defecto. El *<body>*, por tanto, hereda esta característica a menos que definamos otra medida.

Según lo dicho, 1em = 16px, 0.5em = 8px, 10em = 160px etc.

Podemos especificar el tamaño de **cualquier** elemento que necesitemos usando "*em*" **(¡oh!).**

Otro ejemplo sobre la etiqueta H1. Si sabemos que el tamaño estándar del navegador es 16px y queremos que H1 tenga 24px, el cálculo sería: **24 / 16 = 1.5em**

Por supuesto que no todo es "color de rosa"… y sino prueben el Internet Explorer, con seguridad deberán hacerle algún "*hack CSS*" **(¡oops!)**.

Cuál framework escoger

Solo voy a mencionar a modo de ejemplo el caso de *Simplegrid* nombrado más arriba. Este entorno llama a sus espacios "*slots*" y la gama de resoluciones en las que se puede utilizar es amplia aunque dichos "slots" son limitados. *Bootstrap* de *Twitter* es, en ese sentido, más completo y mejor a la vez que más complejo.

Cada framework tiene sus particularidades, sus limitaciones y sus ventajas, y el que vayamos a escoger deberá ajustarse a las necesidades de nuestro "*mock-up*" de modo que juegue a nuestro favor y no se convierta en una piedra a romper. Debemos, por tanto, conocerlos en profundidad, experimentarlos, probarlos, degustarlos, en caso contrario hacernos uno a medida.

Imágenes y medios flexibles

Este tema me trae al presente largas discusiones en foros y con clientes cuyas webs mantenidas o alimentadas de contenidos por ellos mismos se iban poniendo "*pesadas*" y luego venían los reclamos.

Es realmente pasmoso ver que se ha colocado en un artículo cuyo tamaño es de 640px ancho una imagen de 1800px x 600px reducida vía HTML con el atributo "*width*" o escalada "*a mano*". Evidentemente la carga de dicho contenido es lenta y si en la página completa hay varias de ellas, mejor hacerse un café y tomarlo mientras se espera.

Por suerte para el caso de los *frameworks* como *Joomla*[24] existen plugins que, javascript mediante, hacen un excelente trabajo de redimensionamiento y sorteamos estos problemas sacrificando un poco el tiempo del procesador.

De todas formas, siempre es conveniente darle un tratamiento particular a las imágenes y he aquí algunas técnicas para conseguir buenos resultados:

- **Escalamiento CSS**
- **Recorte CSS**
- **Composición con capas**
- **Javascript**

Escalamiento

Esta técnica consiste en asignarle un tamaño por defecto a la etiqueta HTML "*img*" vía CSS, según sean los elementos parientes, y un tamaño máximo.

Ejemplo:

```
<img src="mandrake.jpg" alt="mago mandrake">
```

En el CSS maquetaremos de esta forma:

```
.img{
        width: 20em;
        max-width: 500px;
}
```

Si la imagen al escalarse se ve muy pequeña o se torna imperceptible, podemos agregarle un "*min-width*" (ancho mínimo) además del "*max-width*" (ancho máximo).

Recortar parte de la imagen

En este caso, debemos tomar en consideración dos planteamientos, por lo que serán dos formas diferentes de tratar la imagen:

1- Si es "decorativa"
2- Si forma parte del contenido.

Si la **imagen** es solo **decorativa**, lo que se hace es definir un *div* el cual contendrá la imagen como un fondo o *background*. Esto tiene la ventaja que luego serán fácilmente remplazables modificando solo el CSS.

Ejemplo:

```
<div id="background"></div>
```

En el CSS:

```
div#background {
        background: url(mandrake.jpg) no-repeat;
        width: 50%;
        height: 330px;
}
```

Observen que he dado un ancho y un alto al *div* ya que no tiene contenido, de no hacerlo no se vería nada. Y se define un "*width*" fluido de modo pueda cambiar fácilmente al variar la resolución de pantalla. A la vez se establece un "*height*" fijo para que siempre se vea lo mismo verticalmente.

Si el recorte de la imagen pudiera quedar mejor hacia la derecha, pondremos el código CSS así:

```
div#background {
        background: url(mandrake.jpg) no-repeat right;
        width: 50%;
        height: 330px;
}
```

Si la **imagen** a mostrar pertenece o forma parte del **contenido** este método no es viable, sobre todo si queremos que dicha imagen tenga un enlace, una etiqueta "*alt*" y aparezca en primer plano.

Entonces, debemos hacerlo de esta forma:

```
<div id="foreground">
        <img src="mandrake.jpg" alt="Mago Mandrake" width="500" height="330">
</ div>
```

El código CSS sería:

```
div#foreground {
        overflow: hidden;
        width: 50%;
        height: 330px;
}
```

Observen el uso de "***overflow: hidden***", el cual servirá para ocultar parte de la imagen. Luego iría:

```
div#foreground img {
        float: right;
}
```

Con lo que la porción izquierda de la imagen será visible.

Composición con capas

Es la técnica ideal para diseños fluidos pero la más compleja puesto que las imágenes a utilizar merecerán un tratamiento previo.

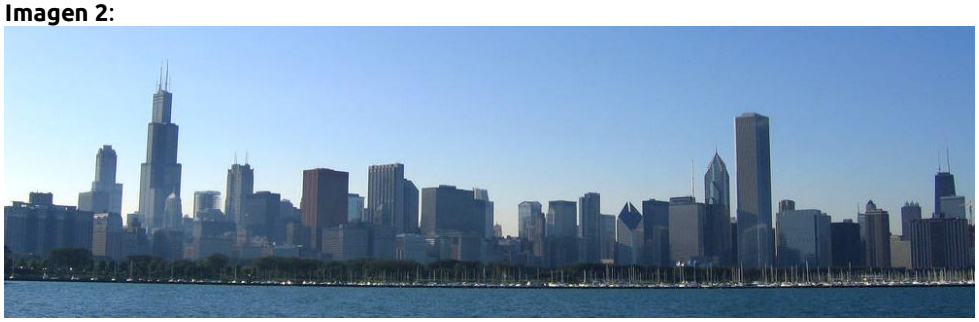
El objetivo buscado es dar apariencia de una sola imagen siendo en realidad una composición formada por varias capas de imágenes superpuestas.

Para ello tendremos que contar con al menos una imagen en formato transparencia (PNG) y superponerla encima de otra. Luego realizar una maquetación CSS utilizando tamaños variables y posiciones absolutas.

Tomemos estas dos imágenes para ejemplo:

Imagen 1 (le he puesto un fondo negro para que se note aunque en realidad es transparente):

Imagen 2:



El código HTML sería:

```
<div id="outer"><div id="inner"></div></div>
```

La maquetación CSS sería:

```
#outer {
        width: 100%;
        max-width: 1000px;
        height: 300px;
        background: url(skyline.jpg) no-repeat;
        }
#inner {
        width: 100px;
        height: 250px;
        background: url(ufo.png) no-repeat;
        }
#inner {
    position: absolute;
    top: 50px;
    right: 50px;
    width: 100px;
    height: 250px;
    background: url(ufo.png) no-repeat;
    }
```

El resultado en el navegador sería este:

Javascript

Quizás esta sea la técnica más utilizada y nuevamente *Ethan Marcotte* viene a nuestra ayuda brindándonos un javascript con este fin.

Podemos leer el artículo respectivo, estudiar el código, incluso descargarlo o copiar y pegar para utilizarlo y probarlo, accediendo a esta URL:
http://unstoppablerobotninja.com/entry/fluid-images/

Otro es *Teleport*, un plugin basado en jQuery cuya url es:
http://teleportz.com.ar/article/plugin-jquery-auto-redimensionar-imagenes

Media queries

Haciendo un repaso general, recordemos que con el nivel de especificación 2 de *CSS* (CSS2) ya podíamos seleccionar qué hojas de estilo usar dependiendo del tipo de medio (*print*, *screen, braille* y otros más), es decir, contábamos con los "*media types*"[25].

En el nuevo *CSS3* se agregan los "*media queries*" con los que podemos seleccionar las propiedades de estos medios, como la anchura del dispositivo, su resolución u orientación (horizontal o vertical). Los "*media queries*" expanden el rol del atributo media, escalan la misma idea hacia un nivel superior y nos dan un mayor control sobre los estilos a aplicar.

Por ejemplo, con los "***media types***" colocábamos en la cabecera de nuestra web algo como esto (lo escribo estructurado para que se note la diferencia):

```
<link
        rel="stylesheet"
        type="text/css"
        href="style.css"
        media="screen"
/>

<link
        rel="stylesheet"
        type="text/css"
        href="print.style.css"
        media="print"
/>
```

El primero es el código para enlazar un CSS de estilo que nos permita representar la página en la pantalla (*screen* traducido es pantalla).

El segundo, enlaza un archivo de hojas de estilo en cascada para poder imprimir la misma página, básicamente simplificándola y/o ajustando márgenes y otras cuestiones.

Con los "***media querys***" podemos hacer lo mismo pero con "vitaminas", algo como esto:

```
<link
        rel="stylesheet"
        type="text/css"
        media="screen and (max-device-width: 480px)"
        href="medium.style.css"
/>
```

Observen que le he asignado al medio "*screen*" un ancho máximo de 480px (max-device-width).

Podría poner muchos ejemplos más, incluso pueden construir códigos según necesiten, pero sino quieren volver a "*reinventar la rueda*", accedan a este sitio:
http://www.stuffandnonsense.co.uk/blog/about/hardboiled_css3_media_queries/

Aquí encontrarán casi todo hecho y si quieren ejemplos de "media queries" accedan a:
http://mediaqueri.es/

Conclusiones y un corolario

Sin dudas que la mayor virtud del RWD es la **navegabilidad** en su máximo esplendor dadas las tecnologías existentes y actuales.

Lo demás dependerá de nosotros desarrolladores porque si bien esta acción del usuario de "navegar" es quizás una de las más importantes, no lo es todo en la conformación de una página o portal web.

Cuando se habla de diseño hay que tomar en consideración otros aspectos y factores que influyen y condicionan sobremanera nuestro trabajo, de los cuales podrá colgar el éxito o el fracaso.

Es común que en charlas de diseño se mencionen palabras técnicas que podemos decir hacen a la coherencia y al sentido común de nuestra tarea. Quizás muchos ignoren estos conceptos pero vale la pena resaltarlos para que luego cada uno las incorpore a conciencia, los tenga en cuenta antes de comenzar la faena, se interiorice de ellos y los aplique, ya que hace no solo al arte del buen diseño web, sino más que nada a hacer las cosas como se debe:

- "Diseño enfocado en el usuario"
- "Navegabilidad"
- "Interactividad"
- "Medios"
- "Usabilidad"
- "Accesibilidad"
- "Funcionalidad"
- "Escalabilidad"
- "Estética"
- "Calidad de la información"

Como ejemplo y a modo ilustrativo del último punto, solo para dar una pista, y tomando las palabras del fundador de "*A List Apart*", Jeffrey Zeldman[26]:

"el contenido precede al diseño. Éste, en la ausencia de contenido, no es diseño, es simple decoración".

Para lo demás investiguen que hay sobrada información por todo el éter.

Corolario:

Hay que tener en cuenta que el diseño responsivo es una tecnología excelente, superior pero reciente, aun está en "*pañales*".

Me ha tocado ver gente utilizando Internet Explorer 6/7 en donde la interpretación de CSS3 por parte estos navegadores es el equivalente a leer de corrido chino básico. Y ¿acaso vamos a dejar "fuera" o podemos ignorar a nuestros usuarios o potenciales clientes por su "*maldito*" programa?

El CSS3 incluso aún no esta soportado en su totalidad por los navegadores más actuales. Hay diversos cuadros en la web donde se indican los porcentajes que cada uno de ellos puede interpretar.

En el mismo sentido, existen muchos problemas de compatibilidad respecto a ciertos códigos javascripts donde algunos navegadores literalmente hacen lo que les viene en gracia y nos dejan totalmente desahuciados, o como mínimo a los insultos.

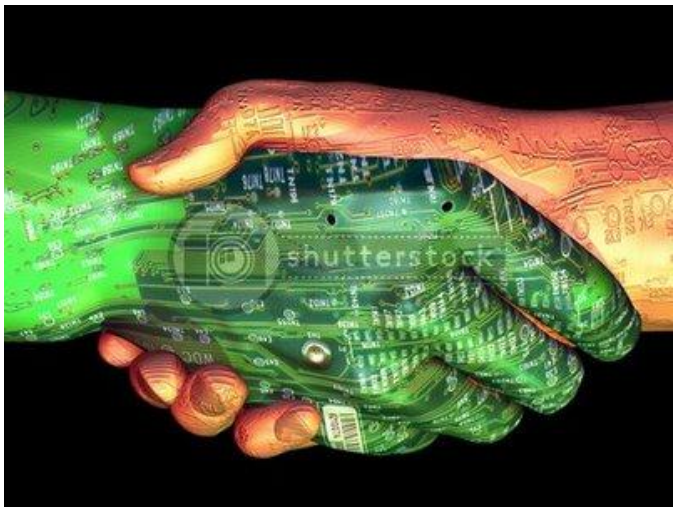
Por más que la W3C trate de "estandarizar", hay firmas, empresas y personas que reniegan de los beneficios que daría un estándar si todos lo siguieran pero es que reina Don Dinero y la "atadura a la marca".

En fin, aunque el diseño responsivo sea una "*tendencia*", aplicarlo o no dependerá de muchas determinantes las cuales surgirán durante las etapas previas de cada proyecto o luego de evaluaciones de proyectos existentes. Se ajustará a cada uno según su necesidad y en vistas a cumplir sus propios objetivos de negocio.

Por otro lado, el uso de RWD supone un cambio de mentalidad respecto a la forma de diseñar una web y quizás no todos estén listos o dispuestos a hacerlo pero tengamos en cuenta que hasta no hace mucho solo el 1% de nuestros visitantes utilizaba un dispositivo móvil y/o una computadora de escritorio, y hoy, los móviles ocupan la nada despreciable cifra del 15%.

¿Hasta dónde aguantaremos el vendaval?

Tecnología del hombre amiga siempre y donde sea



Notas y yerbales

[1] RAE: Real Academia Española
[2] Discusiones sobre la traducción de "responsive": http://joanielena.cat/blog/como-podemos-traducir-responsive-web-design/
[3] Enlace al sitio oficial: http://www.w3.org/
[4] Leyes de Murphy: http://www.joomla-gnu.com/secciones-especiales/arcon-de-la-historia/310-las-leyes-de-murphy.html
[5] Hacks CSS: Los Hacks de CSS son triquiñuelas o trucos para conseguir que una misma declaración de estilos CSS actúe de manera distinta en distintos navegadores. Bajo este concepto se engloban diversas técnicas, muchas veces poco ortodoxas, para lidiar con las distintas interpretaciones de CSS que tienen los distintos navegadores.
[6] Framework: conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar.
[7] Enlace a Blueprint: http://blueprintcss.org/
[8] Web de Ethan Marcotte: http://ethanmarcotte.com/
[9] Enlace al artículo: http://www.alistapart.com/articles/responsive-web-design/ (en inglés) y http://diseñowebresponsivo.com.ar/ (en español)
[10] http://www.abookapart.com/products/responsive-web-design
[11] Enlace al artículo: http://www.alistapart.com/articles/dao/ (en inglés)
[12] Ver: http://vostokproject.com/2011/05/31/arquitecturas-reactivas-y-tecnologias-responsivas-trabajando-desde-el-peor-escenario-posible/
[13] Enlace al artículo: http://www.alistapart.com/articles/fluidgrids/ (en inglés)
[14] Enlace al artículo: http://www.lukew.com/ff/entry.asp?933 (en inglés)
[15] "Conexiones", alusión a la serie Connections realizada por la BBC, creada y conducida por James Burke (leer más aquí: http://es.wikipedia.org/wiki/Connections)
[16] Web de Lucas Wroblewski: http://www.lukew.com/
[17] Biografía y otros artículos: http://www.hesketh.com/author/steven-champeon
[18] Enlace al texto completo: http://es.wikipedia.org/wiki/Mejora_progresiva (en español)
[19] Marcado semántico o semantic mark-up es una técnica de programación HTML que supone utilizar las etiquetas correctas para identificar cada tipo de contenido, se trata de pensar cómo aparecería la web en una página impresa en lugar de como se desea mostrar el contenido en pantalla.
[20] Expertises: término del inglés cuyo significado o definición castellana sería: conjunto de habilidades, conocimientos y experiencias que posee una persona sobre diferentes materias o ciencias. También es común el uso del anglicismo "skills"
[21] Mockup o mock-up: es un anglicismo compuesto cuyo significado es maqueta o esbozo. Al comienzo el término solo era utilizado en ingeniería o diseño industrial y comenzó a aplicarse al diseño web con la aparición de lo que se conoce como Web 2.
[22] Em en W3C: http://www.w3.org/WAI/GL/css2em.htm
[23] Más sobre kerning: http://es.wikipedia.org/wiki/Kerning
[24] Joomla: http://www.joomla.org – CMS (siglas de Content Managetment System, Sistema Gestor de Contenidos) de código abierto con licencia GNU/GPL
[25] Media types CSS2 en W3C: http://www.w3.org/TR/CSS2/media.html
[26] Jeffrey Zeldman en wikipedia: http://en.wikipedia.org/wiki/Jeffrey_Zeldman

Yerbales

Este dossier fue redactado a partir de información leída en la web habiendo algunos enlaces citados en las Notas, también de libros electrónicos y de mi propia experiencia personal en diseño web.

Las imágenes fueron provistas por Google a excepción de los mocku-ps ejemplificados que son fotografías de mis bosquejos tomadas con mi "dispositivo móvil" Samsung Galaxy Y Pro con Android.