

Diseño de Interfaces Web

Unidad 2 (IV)

Diseño “Responsive”



UNIDAD 2 (IV)

Diseño Responsive



1. [Introducción](#)
2. [Viewport](#)
3. [Breakpoints](#)
4. [Media Queries](#)
5. [Patrones responsive](#)
6. [Imágenes responsive](#)
7. [Tablas responsive](#)
8. [Texto responsive](#)
- *. [Referencias](#)

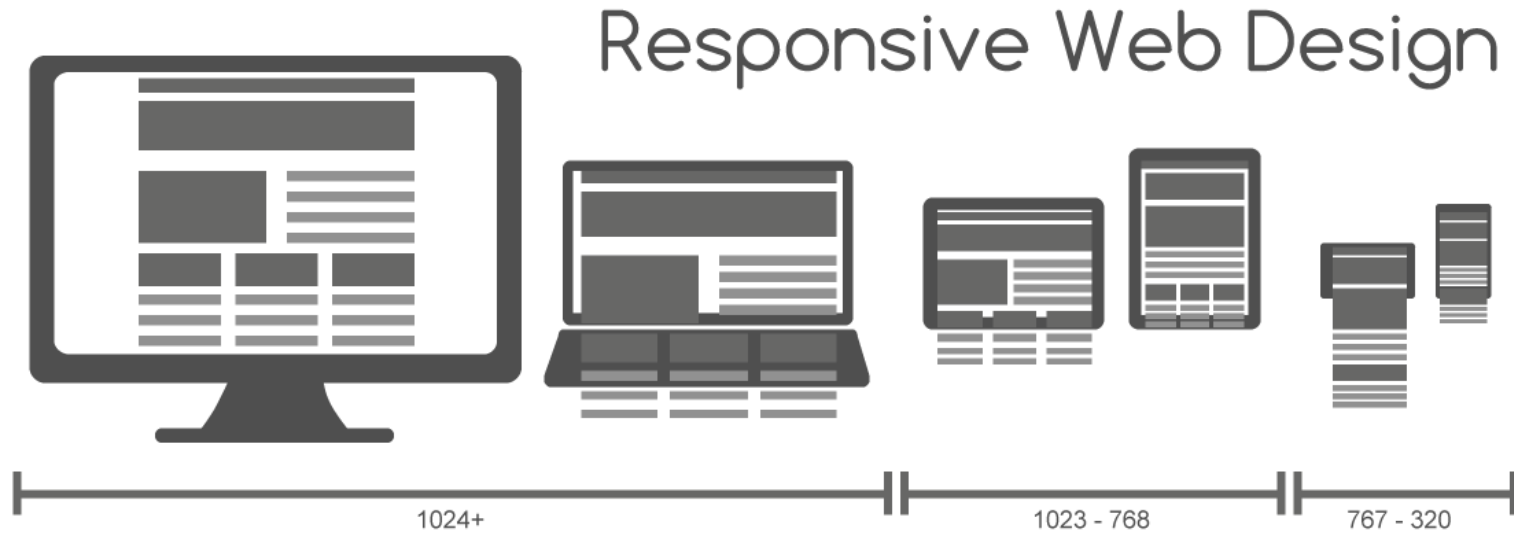


1. Introducción

- ¿Qué es “Responsive Web Design”?
 - **Responsive Web Design (RWD) o diseño “responsivo”** es una tendencia de diseño, en la que se desarrolla una única página que se adapta a los diferentes dispositivos en los que se visualizará
 - Tiene en cuenta tamaños de pantalla, densidad de píxeles, orientación (vertical u horizontal) y elementos de interacción (ratón, pantalla táctil,...)
 - Se desarrolla mediante HTML5 + CSS 3 + *media queries*

1. Introducción

- ¿Qué es “Responsive Web Design”?



1. Introducción

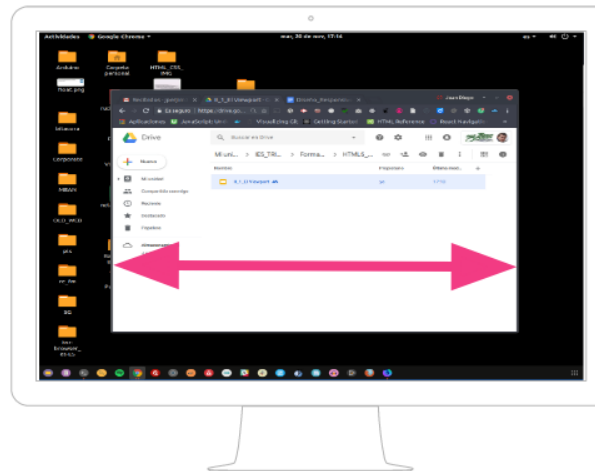
- “Content is like water”





2. Viewport

- Viewport
 - Área visible de una página web (zona en la que el navegador puede renderizar contenido)
 - Cambia según el dispositivo
 - En PC/portátiles coincide con la **ventana del navegador**
 - En móviles y tablets el concepto cambia y hay que hablar de *Layout Viewport* y de *Visual Viewport*



2. Viewport

- Layout Viewport/Visual Viewport (dispositivos móviles)
 - **Layout-viewport:** viewport que es tenido en cuenta para la aplicación de estilos
 - **Visual-viewport:** viewport que realmente ve el usuario cuando está navegando (lo que se renderiza). Hay que tener en cuenta la posibilidad de hacer zoom con los dedos





2. Viewport

- Resolución de pantallas
 - Al existir tantos dispositivos con pantallas diferentes debemos tener en cuenta, además del viewport, otros valores:
 - **Hardware Pixels:** pixeles de resolución que tiene la pantalla
 - **PPI (Pixels Per Inch):** densidad de píxeles (la cantidad de pixeles mostrados por pulgada en la pantalla)
 - **DPI (Device Independent Pixels):** unidad de medida del navegador. Se relaciona con la distancia real y ocupan lo mismo independientemente de la densidad de pixeles de la pantalla
 - **DPR (Device Pixel Ratio):** $\text{HW Pixel} / \text{DPI}$ (es una dimensión)



2. Viewport

- Viewport
 - Un primer paso para poder gestionar toda esta variedad de situaciones y que nuestra página empiece a ser “responsive” es añadir en la cabecera:

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0">
```

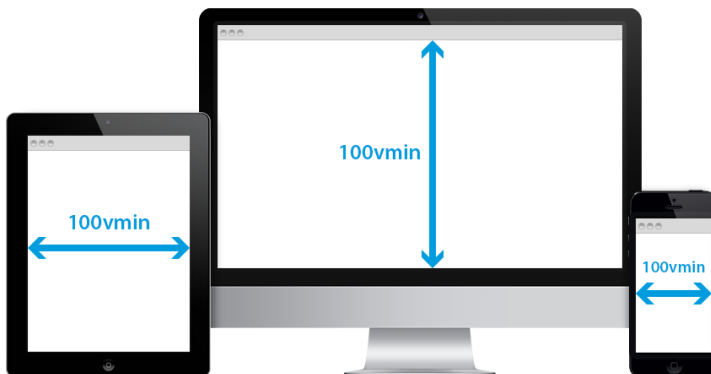
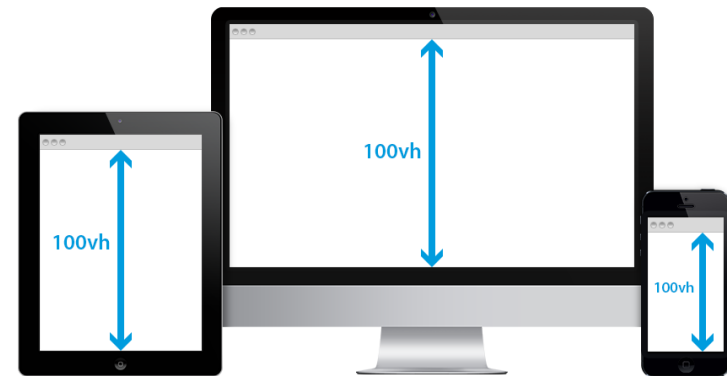
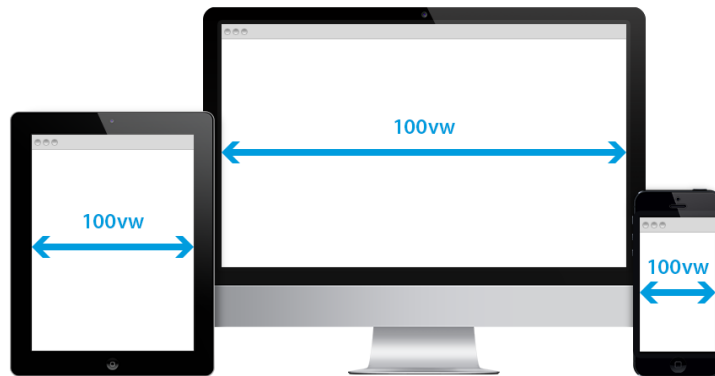


2. Viewport

- Unidades de medida relativas al viewport
 - En HTML5 existen una serie de unidades de tamaño relativas al viewport
 - ✓ **vw (viewport width)**: en relación a la anchura del viewport
 - ✓ **vh (viewport height)**: en relación a la altura del viewport
 - ✓ **vmin**: el valor menor en relación a la dimensión más pequeña del viewport (anchura o altura, según la disposición)
 - ✓ **vmax**: el valor máximo en relación a la dimensión más grande del viewport (anchura o altura, según la disposición)
 - **1vw/1vh** es un 1% de la anchura/altura del viewport
 - Difiere del porcentaje en que no depende del tamaño del contenedor padre

2. Viewport

- Unidades de medida relativas al viewport





3. Breakpoints

- Breakpoints
 - Punto en el que cambian propiedades de una página, generalmente atendiendo a su anchura
 - Ejemplos de breakpoints definidos en *Bootstrap*:
 - < **576px** (pantallas pequeñas)
 - **576px - 768px** (móviles apaisados)
 - **768px - 992px** (tablets)
 - **992px - 1200px** (desktop)
 - > **1200px** (pantallas grandes)



4. Media Queries

- Media Queries
 - Módulo de CSS3 que permite aplicar estilos según el tipo de medio en el que se muestran los documentos
 - A través de la regla `@media` y aplicando distintas condiciones (ancho del medio, orientación, tipo de dispositivo, etc.) se pueden variar los estilos que se utilizan en el documento

```
@media (max-width: 800px)
{
    sidebar {
        display: none;
    }
}
```

```
@media print {
    body {
        font-size: 1.1 em;
    }
    nav {
        display: none;
    }
}
```



4. Media Queries

- Tipos de medios
 - La etiqueta @media reconoce varios tipos de medios predefinidos, al margen de sus propiedades
 - **all** → todos los tipos de medios
 - **screen** → pantallas de ordenadores, tablets, smartphones,...
 - **print** → impresoras
 - **tv** → televisores
 - **tty** → terminales
 - **speech** → utilizado por lectores de pantalla



4. Media Queries

- Condiciones
 - Existe un amplio conjunto de propiedades que se pueden comprobar para aplicar o no determinados estilos
 - *width* | *min-width* | *max-width*
 - *height* | *min-height* | *max-height*
 - *orientation* (*landscape* / *portrait*)
 - *aspect-ratio* | *min-aspect-ratio* | *max-aspect-ratio*
 - *color* | *min-color* | *max-color*



4. Media Queries

- Operadores lógicos
 - Al igual que en los lenguajes de programación, media queries permite utilizar expresiones condicionales mediante los operadores lógicos **and**, or (,), **not**, **only**

```
@media (max-width: 800px), handheld and (orientation: portrait) {  
    ...  
}
```




4. Media Queries

- Ejemplos

```
/* Estilos para todo tipo de pantallas con una anchura máxima  
de 576px*/  
@media all and (max-width: 576px) {  
    .....;  
}  
  
/* Estilos para pantallas con al menos 992px de anchura y que estén apaisadas (más ancho que alto)*/  
@media screen and (min-width: 992px) and (orientation: landscape) {  
    .....;  
}  
  
/* Estilos sólo para pantallas que tengan al menos 768px de anchura*/  
@media only screen and (min-width: 768px) {  
    .....;  
}
```



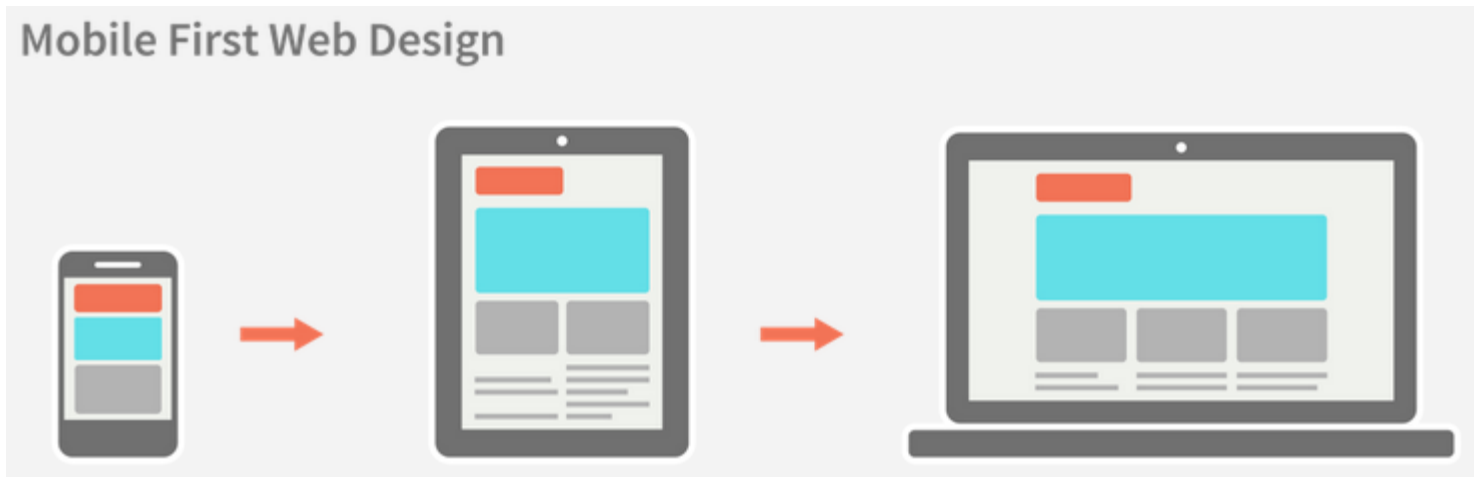
4. Media Queries

- Hojas de estilo diferentes
 - Podemos usar *media queries* en la etiqueta *link* para seleccionar hojas de estilos diferentes según las características del dispositivo

```
<!--Para pantallas de hasta 576px de ancho -->  
<link rel="stylesheet" media="(max-width: 576px)" href="small.css" />  
  
<!--Para pantallas entre 576px y 768px de ancho -->  
<link rel="stylesheet" media="(max-width: 768px)" href="medium.css" />  
  
<!--Para pantallas entre 768px y 992px de ancho -->  
<link rel="stylesheet" media="(max-width: 992px)" href="large.css" />
```

5. Patrones responsive

- Diseño Mobile First
 - Técnica de diseño que consiste en plantear en primer lugar el diseño para dispositivos móviles y a partir de ahí ir desarrollando la interfaz para pantallas más grandes





5. Patrones responsive

- Proceso de diseño Mobile First
 - Al usar *media queries* en diseño *Mobile First* trabajaremos fundamentalmente con la propiedad **min-width**
 - El diseño “Mobile First” es recomendable porque:
 - Se **prioriza** siempre lo que es importante (en un proceso contrario es fácil caer en el error de quitar elementos que son realmente importantes)
 - Nos preguntaremos en cada fase si es necesario un **diseño nuevo** para pantallas más grandes
 - Elegiremos los **breakpoints** más adecuados



5. Patrones responsive

- Diseño Mobile First
 - Patrones Mobile First
 - ✓ **Cabecera no muy grande** para que el contenido no quede muy bajo
 - ✓ **Menús** de navegación **condensados**, desplegándose solo al pulsar
 - ✓ Mostrar los **artículos de arriba a abajo**, ocupando cada uno el 100% del ancho (o el 50% en tablets)
 - ✓ El **contenido lateral no relacionado** con el contenido principal es mejor trasladarlo para que se muestre **en el pie** del documento
 - ✓ Es aconsejable mostrar los **enlaces en forma de botón** con un tamaño fácil de pulsar
 - ✓ Si un artículo se tiene que expandir para ver su contenido ampliado, hacer el título, la imagen y el texto resumen del artículo **“pulsables”** para visitar su versión completa



5. Patrones responsive

- Patrones responsive recomendados
 - **Column Drop**: en cada breakpoint se apila un elemento
 - **Mostly Fluid**: cuadrícula fluida, en cada breakpoint puede haber un redimensionamiento de varias columnas
 - **Layout Shifter**: en cada breakpoint cambia el diseño del layout (no únicamente el flujo y la anchura de los elementos); es el patrón más responsivo
 - **Off Canvas**: en vez de apilar contenidos, éstos se colocan fuera de la pantalla cuando el tamaño de pantalla no es lo suficientemente grande
 - También es posible mezclar varios tipos de patrones o realizar pequeños ajustes (*tiny tweaks*)



6. Imágenes responsive

- Imágenes
 - Las imágenes son un elemento fundamental de todas las páginas y representan una gran parte del **peso** de la misma
 - Esta situación plantea ciertas complicaciones a la hora de hacer diseño responsive
 - Habrá que priorizar entre:
 - **Optimización** de la página web
 - **Diseño** de la página (calidad de las imágenes)



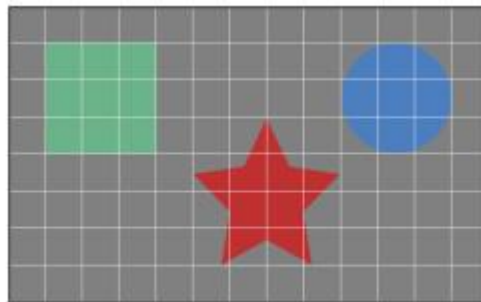
6. Imágenes responsive

- Optimización de imágenes
 - Trataremos de consumir el menor ancho de banda posible
 - Elegir la versión de una misma imagen más adecuada para la resolución con la que estemos trabajando
 - Tendremos en cuenta:
 - Ancho del dispositivo
 - Dimensiones de la imagen
 - Resolución de la imagen (en especial en dispositivos *retina display* -dispositivos con una densidad de píxeles superior a la normal-)

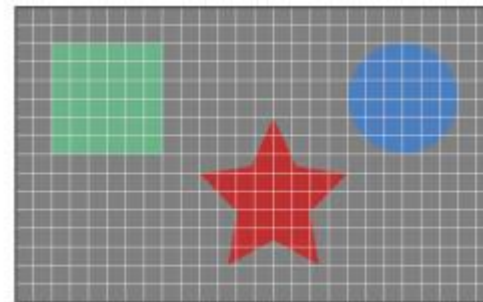


6. Imágenes responsive

- Optimización de imágenes
 - *Retina Display*: en la imagen se muestra el caso de dispositivos con densidad 2x. Para que las imágenes se muestren bien tendrán que ser de tamaño doble



Screen 0



Screen 1



6. Imágenes responsive

- Optimización de imágenes
 - Imágenes SVG
 - Para solucionar los problemas anteriores lo más fácil es usar imágenes **SVG**
 - Son gráficos vectoriales que escalan y encogen sin perder resolución
 - No siempre disponemos de gráficos SVG



6. Imágenes responsive

- Optimización de imágenes
 - Imágenes PNG / GIF / JPEG
 - En estos casos, si no nos importa la optimización es suficiente con usar una imagen de gran resolución y dimensiones y acotarla dentro de un contenedor

```
<div><img src=""....." /></div>
```

```
div {  
    /* dimensiones deseadas */  
}  
  
img {  
    max-width: XXXXXpx;  
    width: 100%;  
}
```



6. Imágenes responsive

- Optimización de imágenes
 - Imágenes PNG / GIF / JPEG
 - Si vamos buscando una optimización de las imágenes haremos uso de los atributos **srcset** y/o **sizes**:

```
<!-- Considerando resolución: -->
<div class="container">
  
</div>

<!-- Considerando dimensiones: -->
<div class="container">
  
</div>
```



6. Imágenes responsive

- Art Director
 - Consiste en elegir una u otra imagen utilizando la etiqueta **source** dentro de **picture** y los atributos **srcset** para indicar la imagen y **media**, que funciona de manera similar a una *media query*
 - Normalmente este diseño consiste en fotos que se acercan al objeto importante

```
<div class="art">
  <picture>
    <source media="(min-width: 576px)" srcset="img/big-art.jpg" />
    <source media="(max-width: 575px)" srcset="img/small-art.jpg" />
    
    <!-- En caso de no soportar picture -->
  </picture>
</div>
```



7. Tablas responsive

- Tablas responsive
 - Las tablas son un elemento problemático a la hora de realizar diseño responsive ya que cuando tienen un número de columnas considerable provocan la aparición de *scroll horizontal* en la página (sobre todo en pantallas pequeñas)
 - Soluciones:
 - Esconder columnas
 - Convertir las filas en listas
 - Crear un scroll horizontal que solo se aplique a la tabla



7. Tablas responsive

- Esconder columnas
 - Se esconden ciertas columnas (las menos importantes) cuando el tamaño de la pantalla es menor que un breakpoint establecido
 - Ventajas
 - ✓ Conseguimos un diseño responsive
 - ✓ Se prioriza el contenido que se quiere mostrar
 - Desventajas
 - ✓ Se pierde información en pantallas pequeñas



7. Tablas responsive

- Convertir filas a listas
 - Se hacen desaparecer las cabeceras de la tabla cuando la pantalla es menor que una determinada resolución y todas las celdas se convierten en elementos de bloque para que se muestren una debajo de otra
 - Ventajas
 - ✓ Conseguimos un diseño responsive
 - ✓ No se pierde información
 - Desventajas
 - ✓ No se prioriza la información
 - ✓ Se desplaza mucho el resto del layout hacia abajo



7. Tablas responsive

- Scroll controlado
 - Consiste en hacer que el scroll horizontal aparezca solo en la tabla (no en la página completa)
 - Ventajas
 - ✓ Conseguimos un diseño responsivo
 - ✓ No se pierde información
 - Desventajas
 - ✓ No se prioriza la información
 - ✓ Sigue habiendo scroll horizontal (solo en una parte)

```
<div class="localscroll">  
  <table>  
    ....  
  </table>  
</div>
```

```
div.localscroll {  
  overflow-x: auto;  
  width: 100%;  
}
```



8. Texto responsive

- Texto responsive
 - Para conseguir un buen diseño responsive, además del layout también debemos controlar el texto de la página
 - Nos podemos encontrar con varios problemas:
 - Líneas cortas con pocos caracteres (dificultan la lectura)
 - Líneas largas con muchas caracteres (también dificulta la lectura)
 - Caracteres muy pequeños que no se pueden leer en pantallas pequeñas



8. Texto responsive

- Texto responsive
 - Encontrar siempre el tamaño ideal de texto es algo que puede ser complicado (si no imposible...)
 - Lo ideal es una letra de tamaño adecuado para no forzar la vista, dispuestas formando líneas de entre 60-80 caracteres
 - Una solución recomendable es utilizar para el texto unidades de tamaño relativas al viewport (*vw*, *vh*, *vmin*, *vmax*)



8. Texto responsive

- Texto responsive
 - Pautas:
 - Calcular el tamaño mínimo y máximo que me puedo permitir (usando la función CSS *calc()*)
 - No importa si el texto hace “reflow” (pasa a la siguiente línea) en pantallas pequeñas
 - Mantener el tamaño perfecto de línea puede que sea imposible
 - Es conveniente **priorizar** el tamaño para **móviles** y **tablets**
 - En algunos elementos, por ejemplo un menú de navegación, puede tener sentido usar tamaño fijos de letra en vez de unidades relativas al viewport



*. Uso de unidades em y de rem

- Unidades em y rem
 - **em** y **rem** son unidades **flexibles**
 - Son unidades escalables que son traducidas por el navegador en valores de pixeles, dependiendo de los ajustes del tamaño de la fuente
 - Aportan flexibilidad a nuestros diseños y la capacidad de que los propios usuarios puedan escalar el tamaño de los elementos
 - La diferencia entre las unidades em y rem es cómo el navegador determina el valor en px al que se están traduciendo



*. Uso de unidades em y de rem

- Unidades em y rem
 - Las unidades **rem** son relativas al valor de fuelle del **elemento raíz de la página** (al elemento **html**)
 - Este tamaño de fuente raíz es multiplicado por el valor de rem usado en nuestro elemento
 - Las unidades **em** son relativas al tamaño fuente del **elemento en el que son utilizadas** (y no, como se cree habitualmente, relativas al valor de su elemento padre)
 - En algunos casos, los tamaños de la fuente del elemento padre pueden afectar a los valores em, pero cuando eso ocurre es solamente **por herencia** (a menos que explícitamente se sobrescriban con una unidad no sujeta a herencia, como px o vw)



*. Referencias

- Bibliografía y referencias
 - Cursos de Openwebinars.net “*Responsive Web Design*” de Juan Diego Pérez
 - Libro “Diseño de Interfaces Web” de Eugenia Pérez Martínez / Pello Xabier Altadill Izura – Ed. Garceta
 - Mozilla Developer Network: <https://developer.mozilla.org/es/>
 - <https://alligator.io/css>