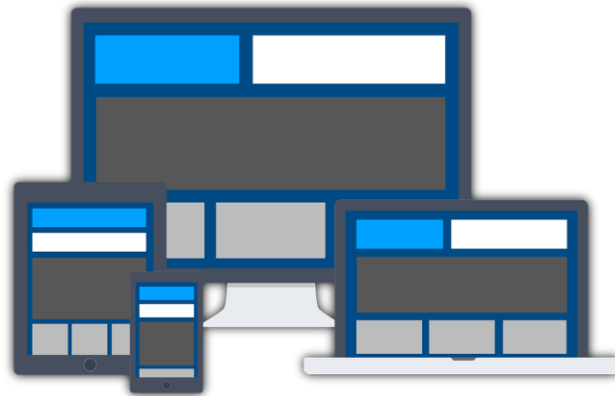


Diseño de Interfaces Web

Diseño “Responsive”





Diseño Responsive

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

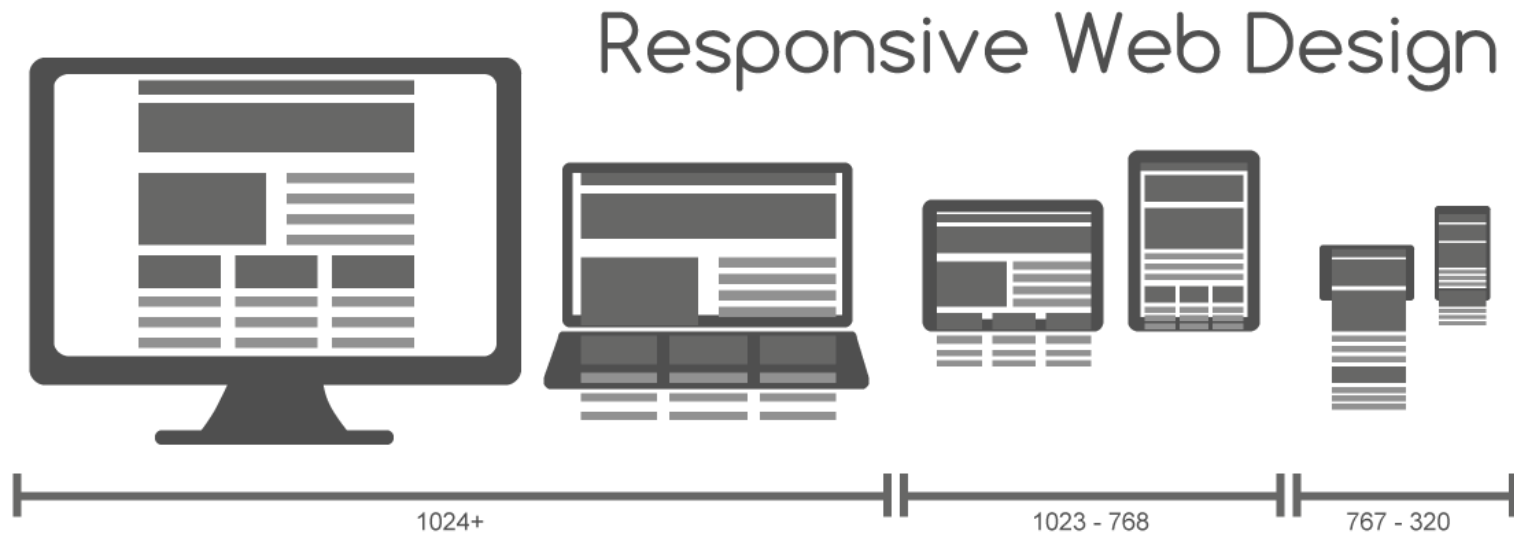


1. Introducción

- ¿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 + CSS3 + *media queries*

1. Introducción

- ¿Qué es “Responsive Web Design”?



1. Introducción

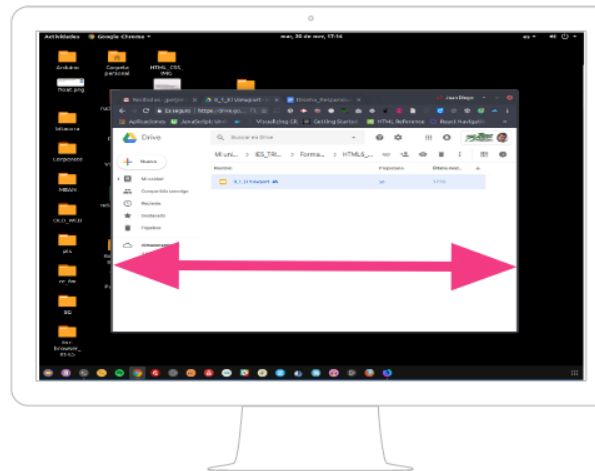
- “Content is like water”





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





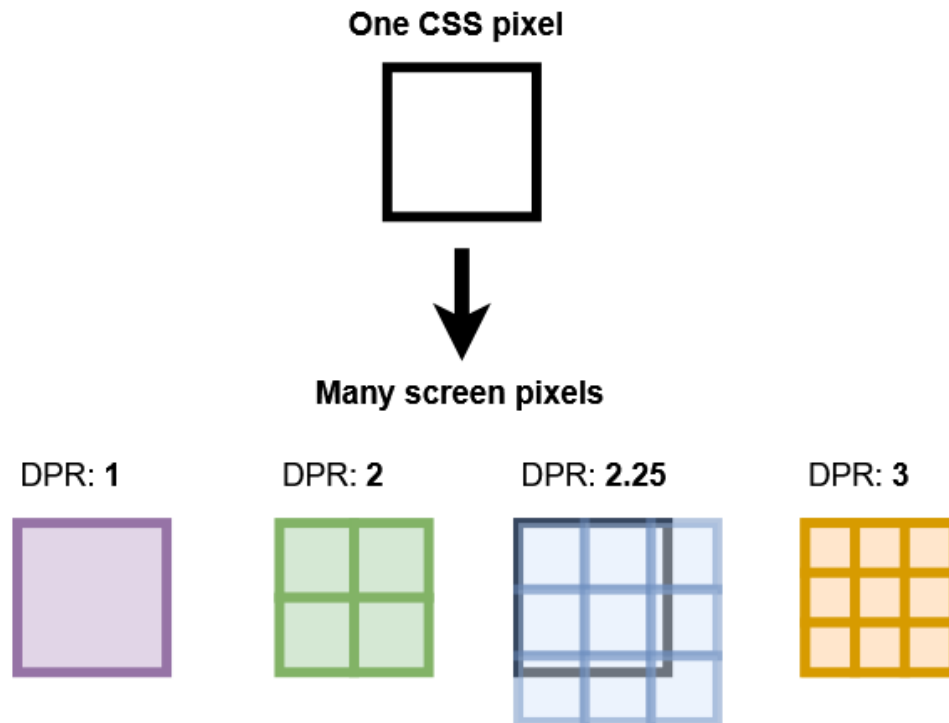
2. Viewport

- 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 (cantidad de pixeles mostrados por pulgada en la pantalla)
 - **CSS Pixels (logical pixels):** pixeles usados al especificar unidades de medida en CSS (puede no coincidir con el número de hardware pixeles de la pantalla del dispositivo)
 - **DPR (Device Pixel Ratio):** número de *hardware pixels* que ocupa un *CSS pixel* en una dimensión



2. Viewport

- Resolución de pantallas
 - DPR





2. Viewport

- Resolución de pantallas
 - DPR
 - La razón por la que se creó el concepto de DPR fue el aumento de la resolución en los dispositivos móviles
 - Con el aumento de la densidad de píxeles físicos, si todos los dispositivos tuvieran un DPR de 1 las páginas se renderizarían demasiado pequeñas
 - Por ejemplo, cuando visualizamos una web en un dispositivo con resolución física de 1080x1920 y DPR de 3, el navegador interpretará que el dispositivo tiene una resolución de 360x640 y las media queries responderán a dicha resolución
 - Pero los elementos renderizados en la pantalla tendrán el triple de definición que una pantalla con una resolución física de 360x640

2. Viewport

- Resolución de pantallas
 - DPR



iPhone 3G

Pixels físicos 320×480

Device Pixel Ratio 1

Pixels lógicos $(320/1) \times (480/1) = 320 \times 480$



iPhone 6

Pixels físicos 750×1334

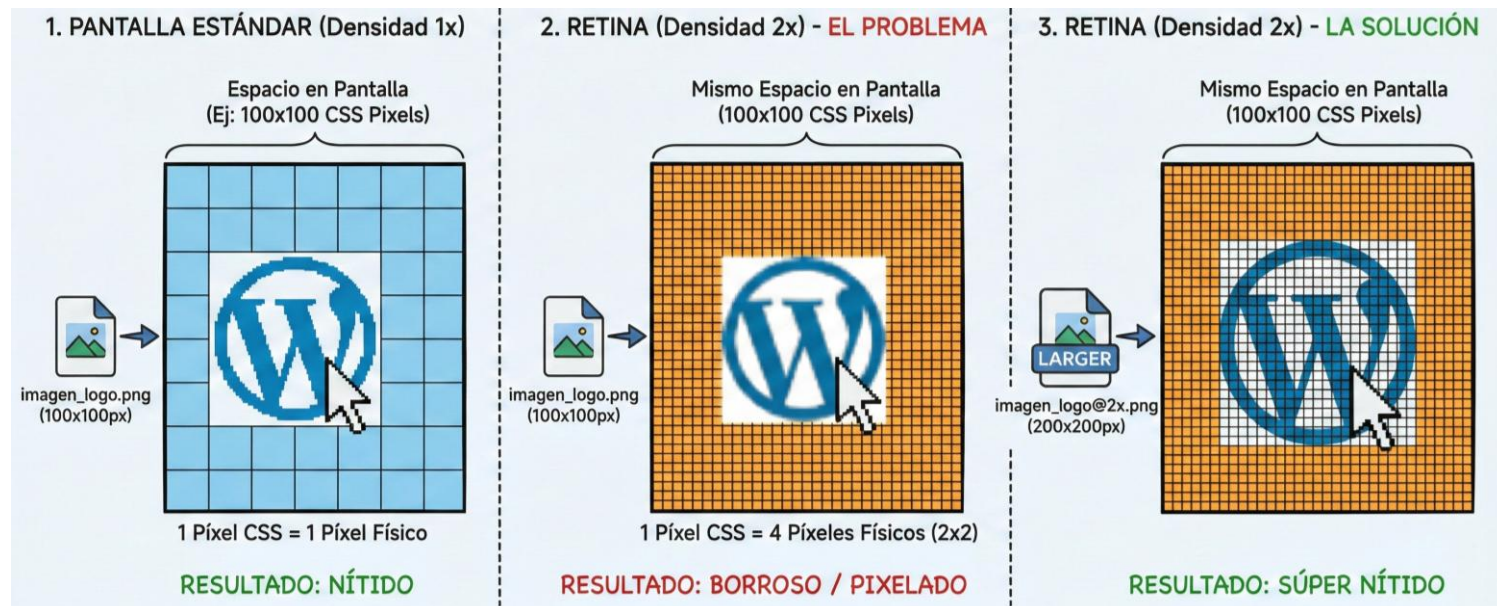
Device Pixel Ratio 2

Pixels lógicos $(750/2) \times (1334/2) = 375 \times 667$

2. Viewport



- Resolución de pantallas
 - En pantallas con DPR mayor que 1 (alta densidad, HiDPI o Retina) necesitamos imágenes mayores para cubrir los píxeles físicos extras y evitar que el navegador la “estire”





2. Viewport

- Viewport
 - Un primer paso para poder gestionar 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">
```

- Se le está indicando al navegador que no importa el dispositivo desde donde se está visualizando la página y que el documento debe ocupar el 100% del ancho de este



2. Viewport

- Etiqueta viewport
 - Valores
 - ✓ **device-width**: ancho del dispositivo
 - ✓ **initial-scale**: valor de zoom inicial
 - ✓ **minimum-scale** y **maximum-scale**: nivel máximo y mínimo del zoom que permitimos
 - ✓ **user-scalable**: define si el usuario podrá o no hacer zoom

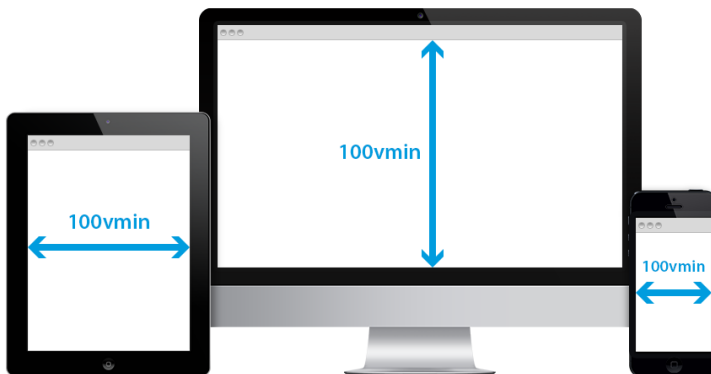
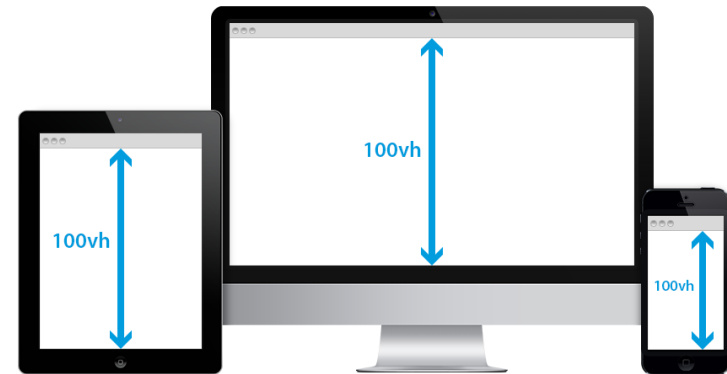


2. Viewport

- Unidades de medida relativas al viewport
 - En CSS3 existen **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 menor valor en relación a la dimensión más pequeña del viewport (anchura o altura, según la disposición)
 - ✓ **vmax**: el mayor valor 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 heredan el tamaño del contenedor padre

2. Viewport

- Unidades de medida relativas al viewport





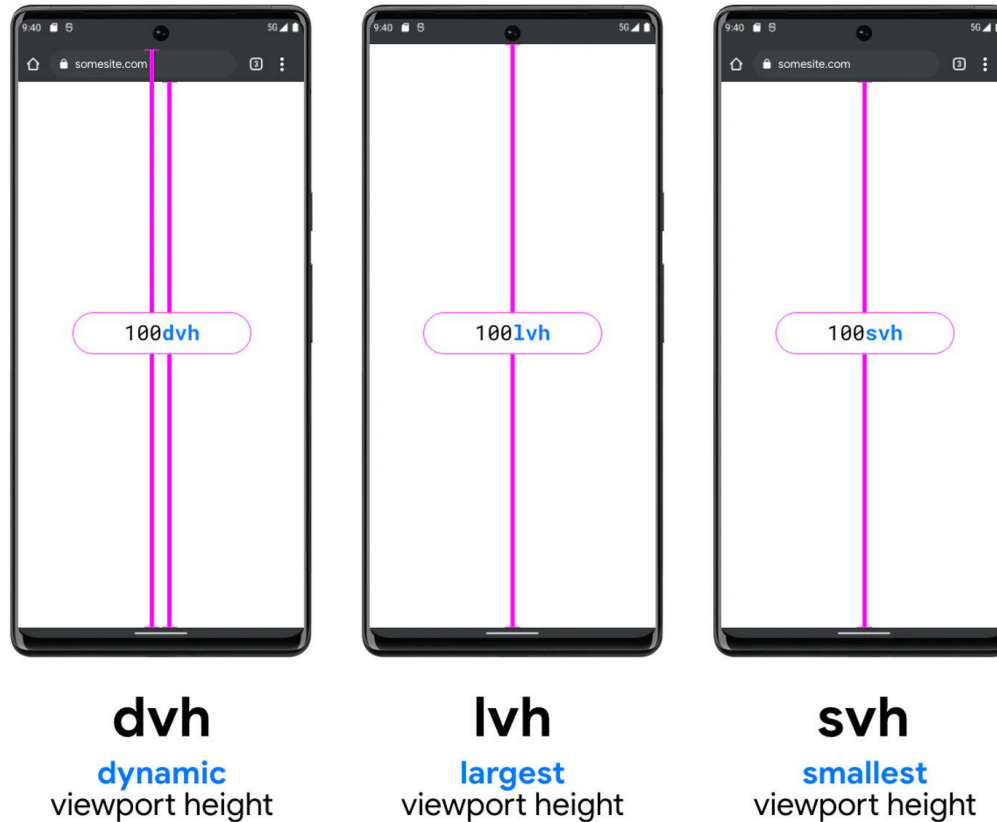
2. Viewport

- Nuevas unidades de medida relativas al viewport
 - Pensadas para solucionar los problemas ocasionados por las barras de navegación y herramientas de los navegadores de dispositivos móviles
 - Estas barras desaparecen al hacer scroll hacia abajo
 - ✓ **svh (small viewport height)** >> altura de la pantalla cuando las barras están visibles (la menor altura disponible)
 - ✓ **lvh (large viewport height)** >> altura de la pantalla cuando las barras se han ocultado o minimizado (toda la altura disponible)
 - ✓ **dvh (dynamic viewport height)** >> unidad “inteligente”: el navegador calcula automáticamente si debe usar svh o lvh dependiendo de cómo estén las barras en ese instante exacto



2. Viewport

- Nuevas 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 screen and (min-width: 900px) {  
  article {  
    padding: 1rem 3rem;  
  }  
}
```



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

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

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



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*
 - También permite utilizar expresiones condicionales mediante los operadores lógicos **and**, **or** (**,**), **not**, **only**

```
@media screen and (min-width: 30em) and (orientation: landscape) {  
    /* ... */  
}
```



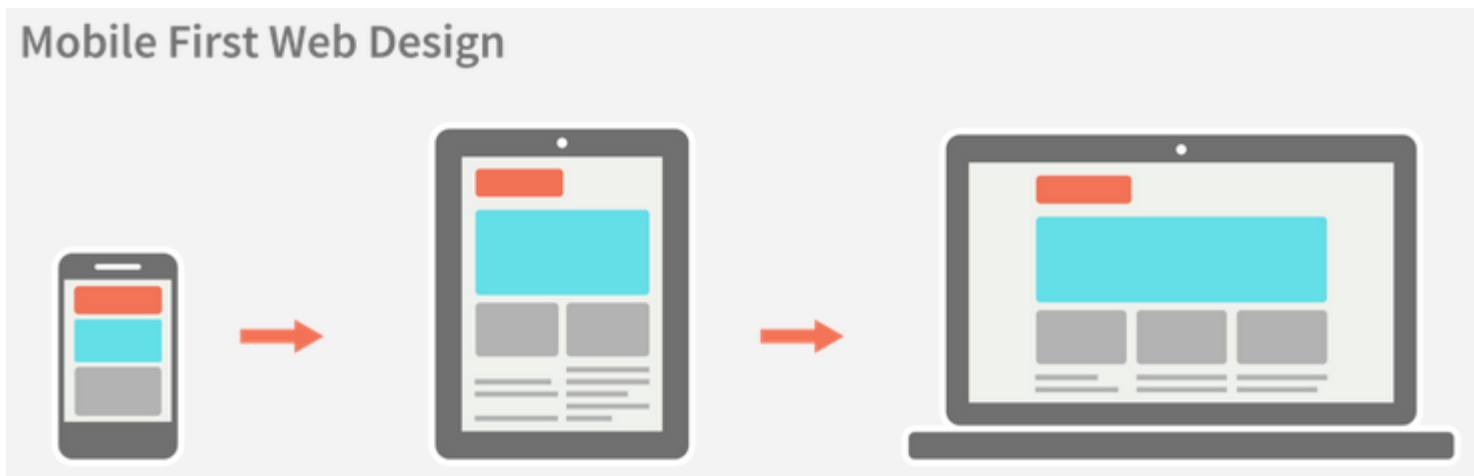
4. Media Queries

- 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
 - El proceso inverso es *Desktop First*





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 *Desktop First* 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 (intentar consumir el menor ancho de banda)
 - **Diseño** de la página (calidad de las imágenes mostradas)



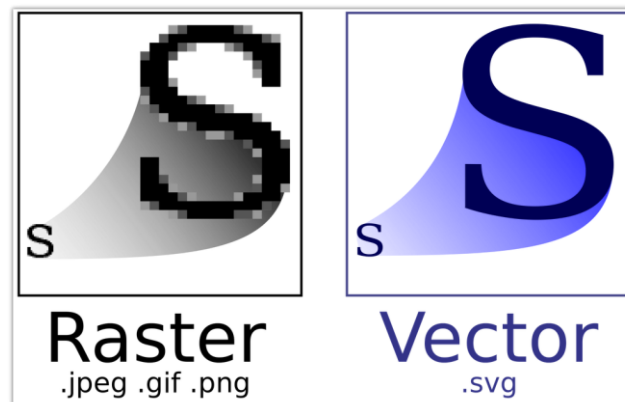
6. Imágenes responsive

- Optimización de imágenes
 - Trataremos de consumir el menor ancho de banda posible
 - Elegir la versión más adecuada de una misma imagen 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 con una densidad de píxeles superior a la normal)
 - DPR del dispositivo: por ejemplo, para un dispositivo con DPR 2, sería conveniente usar imágenes con el doble de tamaño



6. Imágenes responsive

- Formatos
 - Para la optimización lo ideal sería usar imágenes **SVG** ya que al ser gráficos vectoriales escalan y encogen sin perder resolución
 - No siempre disponemos de gráficos SVG (por ejemplo en el caso de fotos), por tanto nos decantaremos inicialmente por formatos clásicos como **JPEG**, **PNG** o **GIF**





6. Imágenes responsive

- Formatos
 - Imágenes WEBP / AVIF
 - Formatos recientes
 - Ofrecen una alta tasa de compresión
 - Aún no tienen soporte en todos los navegadores y requieren algo más procesamiento para su decodificación





6. Imágenes responsive

- Hacer una imagen responsive
 - Como **técnica general** y si no nos importa mucho la optimización, es suficiente con usar una imagen que no tenga unas dimensiones excesivas y acotarla dentro de un contenedor

```
<div class="picture">  
    
</div>
```

```
.picture {  
  width: 400px; /* anchura deseada */  
}  
  
img {  
  max-width: 100%;  
  display: block;  
}
```




6. Imágenes responsive

- Atributos srcset y sizes
 - Si buscamos **mayor optimización** en el uso de imágenes podemos usar los atributos de **srcset** y **sizes** de ****:
 - **srcset**: conjunto de imágenes entre las que el navegador podrá elegir y preinformación de tamaño para el navegador (unidad **w**: anchura)
 - **sizes**: condiciones de medios (ancho de pantalla, DPR,...)
 - Sintaxis: `sizes="[media querie] [length], [media querie] [length],... [default length]"`

```

```



6. Imágenes responsive

- Etiqueta `<picture>`
 - La más reciente etiqueta **<picture>** (que no reemplaza a ``) actúa como un contenedor que permite indicar al navegador qué imagen escoger de un conjunto
 - Dentro de `<picture>` usaremos **<source>** con los atributos **srcset** para indicar la imagen, **sizes** y opcionalmente **media** para el tamaño de pantalla

```
<picture>  
  <source srcset="logo-XL.png" media="(min-width: 800px)">  
  <source srcset="logo-L.png" media="(min-width: 600px)">  
    
</picture>
```



6. Imágenes responsive

- Art Direction
 - Técnica de diseño para composiciones de fotos en la que se enfoca al objeto principal dependiendo del tamaño del viewport
 - Para conseguirlo podemos utilizar **<picture>** para que cargue distintas versiones de la imagen según el viewport

```
<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 responsive
 - ✓ 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 las unidades de tamaño relativas al viewport (vw , vh , $vmin$, $vmax$)
 - De esta forma podemos establecer un valor único para todas las resoluciones de pantalla y que los textos se adapten automáticamente



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



*. 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>