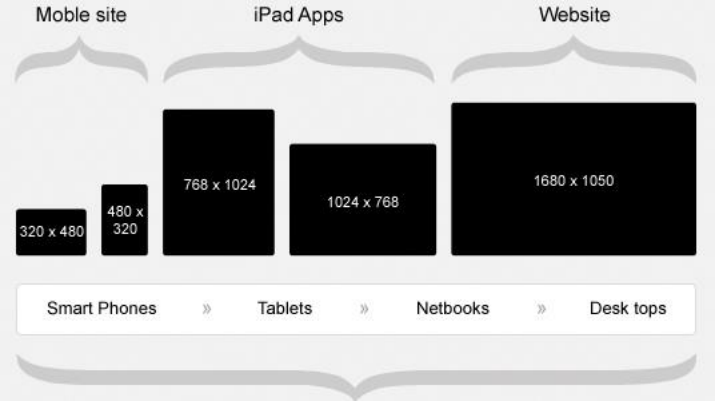
MEDIA QUERIES

Para realizar Responsive Web Design necesitaremos utilizar las media queries de CSS3 ya que éstas nos permiten **especificar estilos condicionales**, aplicables únicamente en determinadas situaciones.

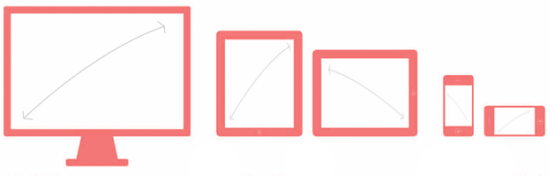
Esta herramienta nos permite **consultar las características del medio** donde se está visualizando la web *(como por ejemplo, la anchura de la ventana)* para así poder aplicar unas reglas de estilos u otras con la finalidad de adaptar la web al dispositivo en el que se está visualizando.

Media Query nos permite establecer **condiciones simples** como las de cualquier lenguaje de programación; "***sí se cumple esta condición, aplica estos estilos***". Algunos ejemplos de uso podrían ser los siguientes:

- Si la pantalla del dispositivo del usuario tiene estas características, aplica estos estilos.
    - *Esto nos vendría bien para poder visualizar correctamente nuestra página web en cualquier tipo de dispositivo con cualquier tamaño de pantalla.*



- Si se imprime el documento en la impresora aplica estos estilos.
    - *Muchas veces es necesario imprimir una página por ejemplo, de confirmación de un pedido o de una reserva. Con esta herramienta garantizamos que los elementos importantes de la página se visualicen correctamente en la copia impresa.*
- Si la pantalla del dispositivo del usuario tiene estas dimensiones y además, está situada en posición landspace (horizontal) entonces aplica estos estilos.
    - *No podemos olvidar que todos los Smartphone y Tablets tienen la opción de landspace luego esto lo tenemos que tener en cuenta en nuestro CSS.*



¿Cómo creamos una Media Query?

Llegado a este punto no debe ser complicado para nosotros crear una Media Query en nuestro fichero CSS. Existen dos formas distintas de insertar medias queries en nuestro diseño web:

- a través de la etiqueta **<link>** haciendo uso del atributo **media** o bien,
- a través de una nueva construcción que comience por **@media** con la sintaxis que veremos a continuación.

Utilizando la etiqueta <link>

Vamos a ver a algunos ejemplos:

```
<link rel="stylesheet" href="estilos-imprimir.css" media="print">
```

Cuando la página se esté mostrando para imprimir se aplicarán los estilos del fichero **estilos-imprimir.css**.

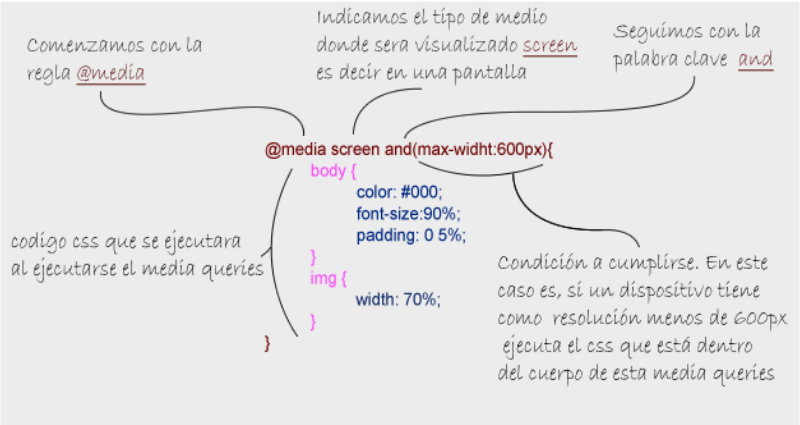
```
<link rel="stylesheet" href="bigscreens.css" media="(min-width:1200px)">
```

Los estilos del fichero **bigscreens.css** sólo se aplicarán cuando la pantalla del usuario tenga una anchura mínima de 1200 píxeles.

El **problema** que tiene esta alternativa es que las hojas de estilos a aplicar se encuentran en ficheros separados por tanto, cuando se vaya a cargar la página se realizará una llamada al servidor para cargar la hoja de estilo en cuestión. Esta técnica ralentizará la carga de la página web. Según esto, **incluir todo el código CSS en un único fichero sería más óptimo**.

Utilizar las sentencias @media

Esta solución es la utilizada a nivel profesional por los diseñadores web. A continuación se muestra un ejemplo de sintaxis de una media query:



Como puedes ver en el ejemplo anterior, las media queries utilizan **operadores lógicos**. Podemos encontrar los siguientes:

- **Operador and**. *Todas las condiciones deben cumplirse para aplicar las reglas de estilo que se encuentran entre llaves.*

```
@media (max-width:600px) and (orientation:landscape){
    h1{color:red;}
}
@media tv and (min-width: 700px) and (orientation: landscape) {
... }
```

- **Operador ,** Este operador equivale al operador lógico **or**, por tanto, el código CSS entre llaves se ejecutará si se cumple al menos una de las condiciones especificadas.

```
@media (min-width:600px), handheld and (orientation:portrait){
    h1{color:green;}
}
```

Este mediaquery servirá para pantallas de anchura mínima 600 píxel y también para todos aquellos dispositivos handheld (agendas electrónicas; no para tablets y smartphones) que estén en posición vertical.

- **Operador not**. Cuando la condición no se cumpla se ejecutará el código CSS. Este operador no se puede indicar para un query individual, sino para un query completo.

```
@media not all and (monochrome) { ... }
```
Equivale a :
```
@media not (all and (monochrome)) { ... }
```

Y no a…
```
@media (not all) and (monochrome) { ... }
```

Del mismo modo,…
```
@media not screen and (color), print and (color)
```
Equivale a `@media (not (screen and (color))), print and (color)`

- **Operador only**. El operador **only** previene que navegadores antiguos que no soportan queries con funciones, apliquen los estilos asignados:

```
<link rel="stylesheet" media="only screen and (color)"
href="Ejemplo.css" />
```

Nota: Ten en cuenta que la mayoría de smartphones simulan tamaños de pantalla mayores, haciendo una especie de dimensiones virtuales que faciliten la lectura de webs que no están diseñadas para Responsive Web Design. Por ello, lo más seguro es que tengas que poner el "viewport" en el documento HTML algo como:

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

Sin ese viewport tu móvil podría simular que tiene unas dimensiones de unos 980píxeles, cuando quizás la pantalla solo tenga una anchura real de 320 o 500píxeles.

Etiqueta meta Viewport

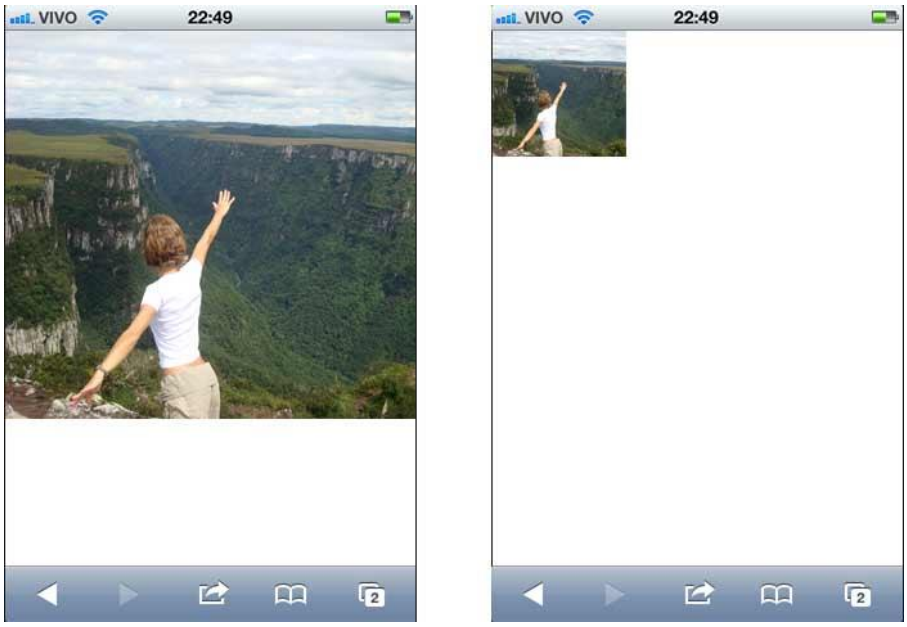
La etiqueta **meta viewport** es una de las etiquetas más representativas de la web móvil, que nos **permite configurar cómo debe interpretar una página, el navegador web para móviles**.

Básicamente, sirve para definir qué área de pantalla está disponible al renderizar un documento, cuál es el nivel de escalado que puede realizar el usuario, así como si el navegador debe mostrarla con algún zoom inicial. Todo ello se indica a través de varios parámetros en la propia etiqueta META.

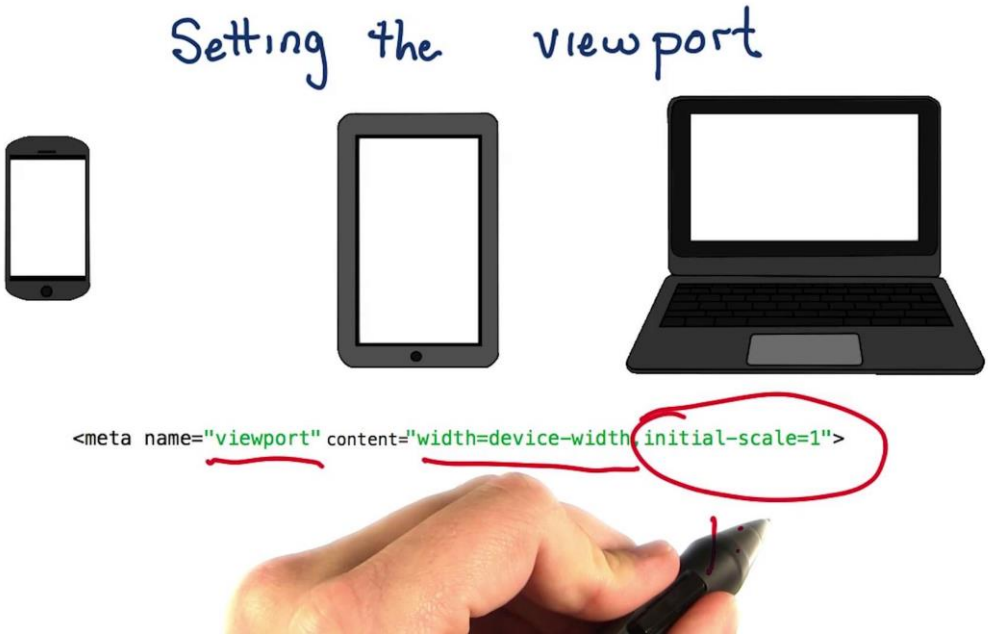
El viewport en un navegador web es el área dónde se mostrará la página web. Si pensamos en el navegador web de un móvil, muchas veces ocurre que el contenido de la web se tiene que reducir de tamaño para poder visualizarse correctamente en la ventana del navegador.

Bien, pues el viewport cuando estamos hablando de dispositivos móviles, no corresponde al tamaño real de la pantalla en píxeles, sino al espacio que la pantalla está emulando que tiene. Por ejemplo, en un iPhone, aunque la pantalla en vertical tiene unas dimensiones de 320píxeles, en realidad el dispositivo está emulando tener 980píxeles. Esto hace que ciertas páginas web (optimizadas para navegadores de escritorio) quepan en una pantalla de 320píxeles, porque en realidad el navegador Safari para iOS está emulando tener un espacio de 980píxeles.

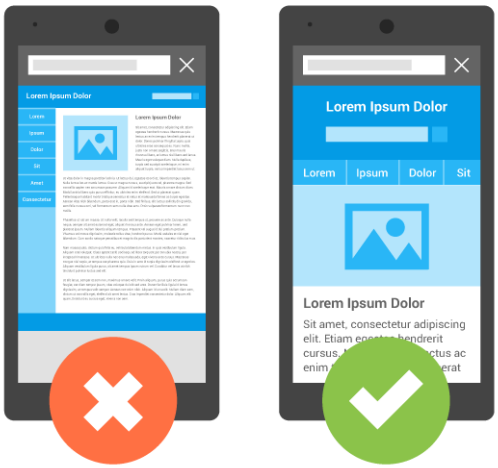
Pues bien, el **viewport** en estos casos **es el espacio que el dispositivo emula tener**, **no la resolución real en píxeles que tiene la pantalla**. Lo interesante en este caso es que los desarrolladores somos capaces de alterar el viewport que viene configurado en el navegador, algo que resulta totalmente necesario si queremos que nuestra página se vea correctamente en dispositivos móviles.



Aquí tenemos la misma fotografía visualizada en dos dispositivos móviles distintos. La de la derecha corresponde a un iPhone. Supongamos que la foto mide 320 píxeles de ancho. En la parte de la izquierda tendríamos la foto a tamaño real, que es como se vería si tuviéramos un viewport configurado a 320píxeles de ancho. Pero al verla en un iPhone con un viewport configurado a 980píxeles de ancho, la imagen se verá bastante más pequeña (imagen de la derecha).



Cuando Safari renderiza un documento web, hace un escalado de los contenidos para que las páginas diseñadas para sistemas de escritorio se vean más o menos bien en un teléfono móvil. Si las escala demasiado habrá que hacer zoom para poder leer el contenido y eso no es interesante. Para evitar esto, nosotros podemos configurar el viewport para decirle al navegador del móvil lo que tiene que hacer.



Ejemplo de viewport:

```
<meta name="viewport" content="user-scalable=no, width=device-width, initial-scale=1">
```

- Con user-scalable=no, conseguimos que el usuario no pueda hacer zoom en la página.
- Con width=device-width conseguimos que el viewport sea igual a la anchura real de la pantalla del dispositivo.
- Con initial-scale=1 conseguimos que no se haga zoom sobre el documento.

Es importante tener en cuenta que **de esta forma estaremos limitando la posibilidad de que el usuario haga zoom puntualmente para agrandar o empequeñecer alguna cosa**. Por todo ello, cabe sopesar bien qué es lo que queremos permitir y si realmente definiendo un maximum-scale y minimum-scale, estamos acotando bien el uso de nuestra web.

Orden de colocación de las media queries

El orden de colocación de las media queries si seguimos la regla de **Mobile First** sería el siguiente:

1. Primero insertar todas las reglas de estilos comunes a todos los dispositivos con CSS3 (sin MQ) y las reglas de estilos que deben afectar a los dispositivos móviles.
2. Después, insertar las MQ necesarias para los dispositivos de siguiente tamaño de pantalla más grande, por ejemplo para tablets.
3. Ir agregando sucesivamente MQ para tamaños de pantalla cada vez mayores.

NOTA: Los atributos **max-device-width** y **min-device-width** no afectan a la anchura de la pantalla actual, sino a la del dispositivo. Para hacer referencia a la anchura de la ventana se utiliza el atributo width con los prefijos min- y max-.

¿Cómo saber el lugar donde colocar un breakpoint?

Para saber dónde colocar los saltos (breakpoints) con las media queries debes estirar la ventana del navegador, partiendo de la ventana con dimensiones reducidas (mobile first) vas estirando la anchura hasta que comienzas a apreciar que tu diseño está viéndose peor.

En realidad, los usuarios no se van a dedicar a cambiar las dimensiones de la ventana del navegador, es algo que hacemos los diseñadores, para imitar rápidamente distintos tamaños de pantalla de dispositivos móviles.

Medidas estándar de pantallas para tus media queries

Como ya se comentó anteriormente, es muy importante tener en cuenta que **el tamaño del viewport no tiene por qué coincidir con el tamaño de la pantalla**, ya que muchos móviles usan varios píxeles de pantalla para representar un píxel de imagen, debido a su **device pixel ratio** o el ratio de pixels que el dispositivo usa para mostrar un pixel en medidas de viewport.

Por ejemplo, la pantalla del Nexus 5 es Full HD, lo que implica una resolución de 1080 x 1920 pixels. Sin embargo, al mostrar una página web el Nexus 5 tiene un viewport de 360 x 598. Todo esto es porque las densidades en píxeles de las pantallas ahora son mayores y para representar un píxel usan varios puntos.

En el artículo Media Queries for Standard Devices de CSS-Tricks, puedes encontrar código de media queries que se podría usar para definir estilos para un dispositivo.

Unidades de medidas CSS (recordatorio)

Unidades fijas

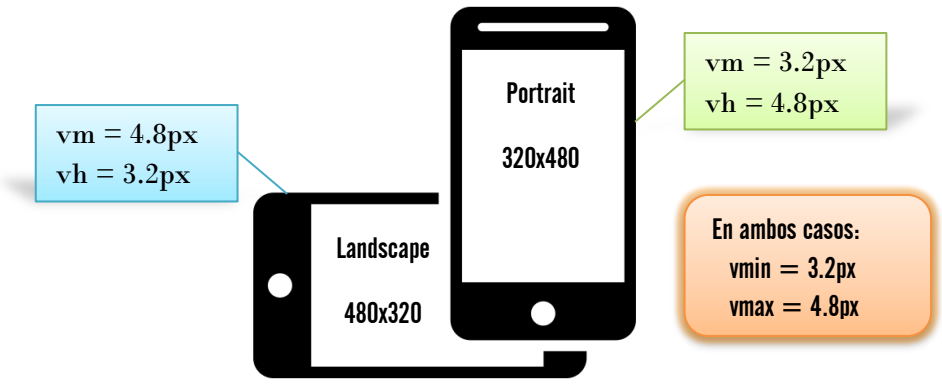
- **px**: Píxeles
- **in**: Pulgadas (1 in es igual a 96px)
- **pt**: Puntos (1 pt es igual a 1/72 in)
- **cm**: Centímetros
- **mm**: Milímetros
- **pc**: Picas

Unidades relativas

- **%**: El porcentaje puede ser relativo frente a varios elementos, si trabajamos con fuentes es relativo a la fuente, pero si lo aplicamos a width es relativo a la anchura del contenedor, por poner dos ejemplos.
- **em**: Relativa al tamaño de la fuente del elemento actual.
- **rem**: Relativa al tamaño de la fuente del elemento raíz (HTML).
- **vw**: Viewport Width, relativa al tamaño del viewport, sería el 1% de la anchura del viewport.
- **vh**: Viewport Height, relativa al tamaño del viewport, sería el 1% de la altura del viewport.
- **vmin**: Relativa al tamaño del viewport, el valor mínimo entre su altura y altura.
- **vmax**: Relativa al tamaño del viewport, el valor máximo entre su altura y altura.
- **ex**: Relativa a la fuente definida en el contenedor, tamaño de la letra "x" en la anchura.
- **ch**: Relativa a la fuente, igual que ex pero con la anchura del carácter 0 (cero).

Aclaración sobre vmin y vmax

Para aclarar **vmin** y **vmax** pensemos en un Viewport de ejemplo de: 320x480 si lo tenemos en vertical (portrait, retrato) y de 480x320 si lo tenemos en horizontal (landscape, paisaje).



Ejemplos de uso de media queries.

Pantalla

```
@media screen { *{ font-family: times; }
```

Impresora

```
@media print { * { font-family: arial;}
```

Pantalla a color

```
<link rel="stylesheet" media="screen and (color)" href="ejemplo.css" />
```

Pantallas con anchura entre 400 y 700px

```
@media screen and (min-width: 400px) and (max-width: 700px) {
        body{    background-color:#555; font-family: Times New Roman;}
        header h3{text-align:center;}
}
```

Pantallas con una anchura de 800px

```
@media screen and (device-width: 800px) { /*Reglas CSS*/ }
```

Pantallas con una altura de 600px

```
@media screen and (device-height: 600px) { /*Reglas CSS*/ }
```

Para todo tipo de medios con una anchura mínima de 500px

```
@media all and (min-width:500px) { … }
```

```
@media (min-width:500px) { … } // Si no indicamos el medio, por defecto es todo.
```

Para todo tipo de medios con una orientación vertical (portratit):

```
@media (orientation: portrait) { … }
```

```
@media all and (orientation: portrait) { … } // Ambas sentencias son equivalentes.
```

Para pantallas con una relación de aspecto determinada:

```
@media screen and (device-aspect-ratio: 16/9) { … }
```

```
@media screen and (device-aspect-ratio: 32/18) { … }
```

```
@media screen and (device-aspect-ratio: 1280/720) { … }
```

```
@media screen and (device-aspect-ratio: 2560/1440) { … }
```

Funciones multimedia

La mayoría de las funciones multimedia pueden ser precedidas por "**min-**" o "**max-**" para expresar "mayor o igual que" o "menor o igual que". *Esto elimina la necesidad de usar los símbolos "<" y ">" los cuales podrían causar conflictos con HTML y XML.*

Ten en cuenta que si usamos una función multimedia sin especificar su valor, la expresión devolverá verdadero si el valor es diferente de cero.

color

Indica el **número de bits por componente** (RGB) de color del dispositivo de salida (`media/visual`). Si el dispositivo no soporta colores, este valor es 0. Acepta los prefijos **min-** y **max-**.

Ejemplos:

Para aplicar una hoja de estilo a todos los dispositivos que soporten colores:

```
@media all and (color) { ... }
```

Para aplicar una hoja de estilo a dispositivos con al menos 4 bits por componente de color:

```
@media all and (min-color: 4) { ... }
```

color-index

Indica el número de entradas en la tabla de colores para el dispositivo de salida (`media/visual`). Acepta los prefijos **min-** y **max-**.

Ejemplos

Para aplicar una hoja de estilo a todos los dispositivos utilizando índices de color:

```
@media all and (color-index) { ... }
```

Para aplicar una hoja de estilo a un dispositivo con un índice de al menos 256 colores:

```
<link rel="stylesheet" media="all and (min-color-index: 256)"
href="http://foo.bar.com/stylesheet.css" />
```

aspect-ratio

Describe el aspecto de una zona a mostrar en el dispositivo de salida (`media/visual`, `media/tactile`). Esto representa la razón de aspecto de los pixeles horizontales a los pixeles verticales (h/v). Acepta los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos cuando la proporción de aspecto de la pantalla del dispositivo de salida sea al 1:1 o superior (cuando el área de visualización sea cuadrada u horizontal):

```
@media screen and (min-aspect-ratio: 1/1) { ... }
```

device-aspect-ratio

Describe la proporción de aspecto en el dispositivo de salida (`media/visual`, `media/tactile`). Esto representa la razón de aspecto de los pixeles horizontales a los pixeles verticales (h/v). Acepta los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos en pantallas con una proporción de aspecto de 16:9 o 16:10:

```
@media screen and (device-aspect-ratio: 16/9), screen and (device-
aspect-ratio: 16/10) { ... }
```

device-height

Describe la altura del dispositivo de salida (`media/visual`, `media/tactile`). Acepta los prefijos **min-** y **max-**.

device-width

Describe la anchura del dispositivo de salida (`media/visual`, `media/tactile`). Acepta los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos cuando un documento sea mostrado en una pantalla de al menos 800px de ancho:

```
<link rel="stylesheet" media="screen and (max-device-width: 799px)" />
```

grid

Determina cuando el dispositivo de salida es un dispositivo de cuadrícula o de mapa de bits. Si el dispositivo está basado en una cuadrícula *(como una terminal TTY o una pantalla de teléfono de solo texto)*, el valor será 1, de lo contrario será 0. No acepta los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos cuando se trate de un dispositivo portátil con una pantalla de como mucho 15 caracteres.

```
@media handheld and (grid) and (max-width: 15em) { ... }
```

height

La función height describe la altura de la superficie a renderizar en el dispositivo de salida (`media/visual`, `media/tactile`) (como la altura de una ventana o la bandeja de papel en una impresora). Acepta los prefijos **min-** y **max-**.

monochrome

Indica el número de bits por pixel en un dispositivo monocromático es decir, en escala de grises (`media/visual`). Si el dispositivo no es monocromático el valor será 0. Acepta los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos para todos los dispositivos monocromáticos:

```
@media all and (monochrome) { ... }
```

Aplicar una serie de estilos a todos los dispositivos monocromáticos con al menos 8 bits por píxel:

```
@media all and (min-monochrome: 8) { ... }
```

orientation

Indica cuando el dispositivo está en modo landscape (horizontal) o en modo portrait (vertical). No admite los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos solo cuando la orientación sea vertical (portrait):

```
@media all and (orientation: portrait) { ... }
```

resolution

Indica la resolución (densidad de píxeles) del dispositivo de salida (bitmap). La resolución puede ser especificada en puntos por pulgada (dpi) o en puntos por centímetros (dpcm). Admite los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos a dispositivos con al menos 300dpi de resolución:

```
@media print and (min-resolution: 300dpi) { ... }
```

scan

Describe el modo en el que un dispositivo refresca su pantalla. Si es progresivo no se aprecian parpadeos cosa que no pasa con un modo entrelazado.
No admite los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos a televisiones de exploración progresiva:

```
@media tv and (scan: progressive) { ... }
```

width

La función width describe el ancho de la superficie a renderizar en el dispositivo de salida (`media/visual`, `media/tactile`) (como el ancho de una ventana de un documento o el ancho de la bandeja de papel en una impresora). Admite los prefijos **min-** y **max-**.

Ejemplo

Aplicar una serie de estilos a dispositivos portátiles o pantallas con un ancho de al menos 20em, usted puede usar esta query:

```
@media handheld and (min-width: 20em), screen and (min-width: 20em)
{ ... }
```

Esta query especifica una hoja de estilo para ser aplicada a un medio impreso con un ancho mayor a 8.5 pulgadas:

```
<link rel="stylesheet" media="print and (min-width: 8.5in)"
href="http://foo.com/mystyle.css" />
```

Esta query especifica una hoja de estilo para ser utilizada cuando la ventana tiene un ancho entre 500 y 800 pixeles:

```
@media screen and (min-width: 500px) and (max-width: 800px) { ... }
```