Kseso — 3.6.15 — 4 Comentarios— Seleccionar idioma ▼ —Tags:demo · Picture element · responsive image · RWD—Rev: 2015-11

> Responsive images: Una guía práctica para comprender y usar los distintos elementos y atributos que permiten utilizar imágenes "responsive" en la web actual y los atributos y valores para seleccionar en base a distintos criterios: art, dpi, tamaños...

> Aprovecho que Firefox 35 (Junio 2015) también lo soporta para marcar este artículo como postCrossing. Haz con él lo que quieras.

El elemento `picture` y su hijo asociado `source` junto a los atributos `srcset`, `sizes`, definidos originalmente para el elemento `img`, es la respuesta actual vencedora en la batalla para dotar al desarrollador de herramientas para el manejo de las imágenes en el escenario del*responsive web design*.

Así, con un uso correcto de estos elementos y atributos, ya es posible disponer de *responsive images*: servir el archivo de imagen que mejor se adapta al dispositivo que recibe la página.

Origen imagen: Internet Archive Book Images

El elemento picture

Del elemento `picture` dice la especificación que es un contenedor que provee múltiples orígenes para la imagen contenida en él. Lo que permite a los desarrolladores controlar o sugerir al navegador qué imagen mostrar.

El elemento `picture` debe tener dos hijos:

- Un elemento `source` como mínimo
- Un elemento `img` obligatorio

El valor del atributo `srcset` es una ruta a un recurso de imagen o un grupo de rutas separadas por comas. Cada valor de ruta y separado por un espacio en blanco puede ir acompañado de algún tipo de información o pista para que el navegador seleccione la que mostrar en base al tamaño del viewport o la densidad del dispositivo.

El elemento `img` es *requerido*. Esto es, si no se incluye no se mostrará ninguna imagen. Su ausencia convierte al

elemento `picture` en un elemento vacío.

El elemento `source` contiene los grupos de imágenes entre los que se elegirá la que mostrar. Un mismo `picture` puede contener tantos elementos `source` como se necesiten.

El atributo `srcset` también puede ser utilizado en los elementos `img` (también si img es hijo de un `picture`). En cuyo caso es obligado incluir además del *srcset* el atributo `src`:

Criterios de selección de la imagen

Bien, gracias al elemento `source` indicamos al navegador el paquete de imágenes disponibles para elegir una de entre todas. Pero por sí mismo poco o ningún control hay sobre la imagen final mostrada.

En estos momentos, hay cuatro criterios de selección que se identifican con cuatro palabras claves (o keywords): **size, dpi, mime y art** en base a ellos, el desarrollador puede sugerirle al navegador qué imagen mostrar.

size
: Por tamaño del viewport

dpi
: por densidad de píxeles del dispositivo

mime

por tipo del archivo de imagen (extensión)

art

por criterios artísticos en función de la imagen y contextos de presentación

Ahora la cuestión es ¿cómo indicarle ésto al navegador? Y la respuesta es sencilla: mediante los distintos atributos disponibles para ello: **sizes, media, type** (son opcionales) junto al ya indicado**srcset**.

Estos criterios se pueden usar individualmente o de forma conjunta varios de ellos.

Selección por densidad de píxeles: DPI

Para comenzar, un caso sencillo. Tengamos el siguiente código. Uso el elemento `img` para hacer más sencillo el código. Pero es igual si el elemento utilizado fuese un `picture`.

En el valor del atributo `srcset` se indican las rutas a tres imágenes. A continuación cada ruta (separado por un espacio en blanco y antes de la coma que las separa) se indica un valor *Nx* que es la referencia para que el navegador seleccione la que mejor se ajuste a la densidad de píxeles (DPI) del dispositivo que la mostrará.

Observa que la primera carece de este valor (*Nx*) porque por defecto o en su ausencia el valor es `1x`.

En este caso, los distintos archivos de imagen pueden ofrecer o no la misma visión. Lo que sí es que cada archivo está guardado en una resolución diferente.

Selección por tamaño: atributo sizes

Para mostrar imágenes en función del tamaño de la pantalla o ventana se utiliza el atributo `sizes=''`. Admite un valor `sizes='100vw'` o múltiples valores separados por comas.

Los valores admitidos como válidos son cualquiera de los usados en tamaños (px, rem, unidades relativas al viewport) e incluso @medias queries: `sizes="(max-width: 30em) 100vw, (max-width: 50em) 50vw, calc(33vw - 100px)"`

En este último código de ejemplo el valor que precede a la coma indica la superficie que cubrirá la imagen en cada tamaño de ventana. El último es el usado como valor por defecto.

```
Con este código, en ventanas de 800px o mayores la imagen se muestra ocupando el 60% de la anchura de la ventana y en menores el 100%. Y en base a ello el navegador mostrará una de las imágenes indicadas en el srcset en función del tamaño de la ventana y de la densidad de píxeles de la pantalla.
```

También se puede indicar archivo de imagen por tamaño de ventana de la siguiente forma con el elemento `source` y el atributo `media`:

Por formato de archivo

En caso de querer ofrecer diferentes tipos (extensión) de archivo de imagen, el atributo a utilizar es `type`.

Los navegadores que soporten el formato*webp* mostrarán dicho archivo, el resto el jpg.

Por criterios artísticos: art direction

Debido a la disparidad que hay en la relación de aspectos de pantallas, hay imágenes que por su*composición artística* perderían la información relevante si sólo se escalasen en su tamaño para ajustarse al tamaño de las pantallas o su densidad de px. Como en la imagen de abajo.

Imágenes "responsives" sólo por tamaño del viewport

Sin embargo, utlizando criterios artísticos en vez de basarnos en el tamaño del viewport el resultado será

mucho más satisfactorio:

Imágenes "responsive" por criterio artístico

Tengamos dos recortes distintos de una fotografía, uno en vertical y otro en horizontal, para ser mostrado en función de la orientación del dispositivo (landscape o portrait):

Recuerda que como valor del atributo `media` se puede utilizar cualquiera de los admitidos por las `@medias queries` de Css.

Ver Demo en Codepen

Picture por múltiples criterios

Lo que suele ocurrir en la realidad es que las cosas suelen ser más complejas. Así que lo más fácil que te puede ocurrir es que el código final a utilizar en cada caso sea función de las características de cada imagen más aquellos escenarios que necesites o desees cubrir.

Será común que tengas un set de archivos de la misma imagen a distintos tamaños y/o resoluciones y a la vez distintos recortes de ella también en distintos tamaños y resoluciones.

No será extraño, pues, que necesites combinar dos o más criterios: art direction + size + dpi, por ejemplo:

```
En viewports de anchura 1280 píxeles CSS o más, se muestra una imagen completa ocupando un 50% del ancho de la ventana utilizada; para las ventanas del navegador con una anchura de 640-1279 píxeles CSS, la imagen ocupa un 60% del ancho de la ventana; por ventanas del navegador menores a 640px, se usa una foto con una anchura que es igual a la anchura de la ventana gráfica completa.
En cada caso anterior, el navegador mostrará una de las imágenes opcionales indicadas en el atributo `srcset` (cada una de ellas con un ancho de 200px, 400px, 800px, 1200px, 1600px 2000px) teniendo en cuenta el ancho de la imagen y la densidad de la pantalla (DPI).
```

La anchura le es sugerida al navegador con el valor numérico terminado con `w` que acompaña a cada valor o ruta del srcset separado por un espacio en blanco.

En la siguiente demo podrás ver el cambio de imagen según sea la orientación y tamaño de la ventana porque cambia el valor de la cantidad (dimensiones de la imagen) que hay impreso en cada imagen.

En caso contrario te aparecerá el valor del atributo *alt* del elemento `img` del `picture` pues la ruta

de la imagen no apunta a ninguna.

Lecturas complementarias y soporte

Sí. Esta solución y recurso para lidiar con imágenes*responsives* es a la vez potente, versátil y compleja. Y el mejor código (por efectivo y útil) dependerá de cada situación en concreto: tipo de imagen y situaciones a contemplar.

Para dominarlo, nada mejor que escribir código y ver resultados. Pero hoy por hoy no en todos los navegadores verás el funcionamiento tanto del elemento `picture` y sus hijos asociados así como de

los atributos

Artículo en postcrossing

implicados. Y visto el ritmo con el que los equipos de desarrollo sacan versiones nuevas, lo mejor es que en cada momento consultes su soporte:

Consulta soporte

- Soporte picture element
- Soporte atributo srcset

Funcional, para mi y en el momento de escribir el artículo, en Chrome estable y Chrome Canary últimas versiones en windows.