Concepts and examples of responsive images

In working through automaating image comperession and responsive image generation I wanted to add more areas to the conversation about responsive images without cluttering the post about automation.

How should we get our source images?

The first thing to consider is what should be the ideal format for our source images. The Gulp task with manipulate the size of the images so it pays to have the largest possible image available to process. It’s always better to shrink an image than it is to enlarge it, besides you will compress the images later anyways.

There are ways to use Machine Learning to improve the quality of existing images even if we don’t have access to the originals.

Device Pixels versus Logical Pixels

We need to look at the difference between logical pixels and device pixels since this will affect the way we generate our responsive images and what types of images we want to use for our originals.

Let’s assume that we have an iPhone X with a resolution of 2436 x 1125 device pixels and Device Pixel Ratio (DPR) of 3.

> The DPR is defined by the device manufacturer. Simply put, it refers to the number of physical pixels contained in one logical pixel. For example, a device with a DPR of 2 means that one logical pixel contains 4 (2 x 2) physical pixels. Similarly, a DPR of 3 implies that a single logical pixel is equivalent to 9 physical pixels. from A Guide to Responsive Images on the Web

If we divide the height and width by the DPR We get the logical size of the screen in this case: 812 x 375 and that’s why the following CSS would work:

```
.example {
  width: 400px;
}
```

We’re telling the CSS parser to use 400 logical pixels instead of 400 device pixels.

Native Responsive Images

There are two ways to create native responsive images: the `picture element` and the `srcset and sizes attributes`. We’ll look at how each of these methods work and then we’ll look at the use cases.

Picture

Using picture and source elements in your page is similar to the video element. You use one or more source elements to indicate the different images that we want to use.

We can leverage the picture element to work with WebP in browsers that can support those formats.

The following code example gives us the following:

1. If the width of the screen is 40em and the browser **supports** WebP use this source and the corresponding image based on device’s DPR
2. If the width of the screen is 40ems and the browser **does not support WebP** use this source and the corresponding image based on device’s DPR
3. If neither 1 and 2 are met uses this source and the corresponding image based on device’s DPR
4. if the brrowser doesn’t understand the `source` attribute then use the `img` element. This must be the last children of a `picture` element

```
<picture>
  <!--1-->
  <source media="(min-<source media="(min-width: 40em)"
    srcset="big.webp 1x, big-hd.webp 2x"><!--2-->"(min-width: 40em)"
    srcset="big.jpg 1x, big-hd.jpg 2x">
  <!--3-->
  <sourc<!--3--><!--3-->all.jpg 1x, small-hd.jpg 2x">
  <!--4-->
  <img src="fall<!--4--><!--4-->
  <img src="fallback.jpg" alt="">
</picture>
```

srcset and sizes

srcset defines the set of images we want the browser to choose between, and the DPR for each image. The browser will select the best image to use based on the device’s characteristics.

The 2x, 3x and 4x values indicate different DPR values. Where there is no value it is assumed to be 1x.

```
<img
  srcset= 'two.png 2x,
           three.png 3x,
           four.png 4x'
  src="small.jpg" alt="A rad wolf">
```

We can also add a width attribute instead of the DPR indicator. To assign the width of the image, we add a `w` to the pixel width number for the image.

The values for `w` can be almost any length value, e.g. em, rem, pixels, and viewport width, **except percentages**. The vw value is recommended as an alternative if a relative value is needed.

Only when we use width values we can add the sizes attribute.

The `sizes` attribute gives a set of media query-like conditions and width of images to use if the condition is met.

> A media condition is not exactly a media query. It is part of a media query. It doesn’t allow us to specify media types, e.g. screen or print, but accepts the condition we usually add to a media type.

> A valid media condition can be either -
>
> - A plain media condition, e.g. `(min-width: 900px)`
> - A "not" media condition, e.g. `(not (orientation: landscape))` An "and" media condition, e.g. `(orientation: landscape) and (min-width: 900px)`
> - An "or" media condition, e.g. `((orientation: portrait) or (max-width: 500px))`
>
> From: [Responsive Images - The srcset and sizes Attributes]

As I understand it the idea is that the browser will check the media conditions in the `sizes` attribute and, based on what condition is used, the browser will select the image to used from those available in the `srcset`.

The full example using `srcset` and `sizes`

```
<img
  srcset= "large.jpg 1024w,
          medium.jpg 640w,
          small.jpg 320w"
  sizes=  "(min-width: 48em) 33.3vw,
          (max-width: 25em) 75vw,
          100vw"
  src="small.jpg" alt="A rad wolf">
```