IMAGES

Terms

Art direction
Clipping
CSS sprites
Data URLs
Device Pixel Ratio (DPR)
Filters
High-density screens
Icon fonts
Logical resolution
Physical resolution
Raster images
Resolution switching
Retina displays
Scalable Vector Graphics (SVG)
Vector images

Summary

- Images fall into two categories: raster and vector. Raster images are made up of pixels. Vector graphics are defined by a set of mathematical vectors (eg lines and curves).

- Raster images often come from cameras and scanners but they can also be produced in software. Any file with the extension of JPG, PNG, GIF is a raster image. We use these images for displaying photos.

- Vector images are exported from drawing tools like Adobe Illustrator. Files with the SVG extension represent vector graphics.

- We use the `img` element to display content images. Content images can represent meaningful content or be used for decorative purposes. If used for decoration, we should set the `alt` attribute to an empty string; otherwise, screen readers will read out the name of the file which may be distracting to the user.

- Using CSS sprites we can combine multiple images into a single image (sprite) and reduce the number of HTTP requests. The problem with CSS sprites is that every time we need to change one of the images in the sprite, we have to re-generate the sprite. So, use this technique for small images that don't change often. You can generate a sprite using https://csspritestool.com.

- Data URLs allow us to embed an image data directly in an HTML document or a stylesheet. The embedded code is always greater than the size of the original resource and makes the document convoluted and hard to maintain. Use this technique only if you know what you're doing!

- We can clip an image using the `clip-path` property in CSS. To generate a clip path from basic templates, visit https://bennettfeely.com/clippy

- Using the `filter` property in CSS, we can apply filters such as grayscale, blur, saturate, brightness and so on.

- High-density screens like Apple's Retina displays contain more pixels than standard-density screens. The pixels on these screens are smaller than the pixels on standard-density screens. So when displaying an image, the screen uses a scale factor (1.5 or greater) to scale up the image. As a result, raster images may look a bit blurry when shown on these screens. To solve this problem, we can supply 2x or 3x versions of an image using the `srcset` attribute of the `img` element.

- For flexible-sized images, we need to supply the image in various sizes for different devices like mobiles, tablets and desktop computers. If we supply a single image, the browser on each device has to resize the image which can be a costly operation. The larger the image, the more memory is needed and the more costly the resizing operation will be. Plus, the extra bytes used to download the image will be wasted. This is the *resolution switching* problem. To address this, we should give the `img` element a few image sources and the size of the image for various viewports. The browser will take the screen resolution and pixel density into account and download the image that best fits the final size.

- We can use https://responsivebreakpoints.com to generate our image assets for various screen sizes.

- WebP is a modern image format created by Google and is widely supported except in Internet Explorer. To support modern image formats, we can use the `picture` element with multiple sources. The `picture` element should always contain an `img` element otherwise the image is not shown.

- Sometimes we need to show a zoomed in or a cropped version of an image for certain viewport sizes. This is the *art direction* problem. To handle this, we use the `picture` element with multiple sources. Each source should contain a media condition and a `srcset`. The browser will pick the first source whose media condition matches.

- Scalable Vector Graphics (SVG) files are great for logos, icons, simple graphics and backgrounds with patterns. They are often very small and can scale without losing quality. You can get find plenty of beautiful SVG backgrounds on https://svgbackgrounds.com

- We can also use icon fonts for displaying icons. The most popular icon fonts are Font Awesome, Ionicons and Material Design Icons.

CSS Cheat Sheet

Background Images

```
background: url(../images/bg.jpg);
background-repeat: no-repeat;
background-position: 100px 100px;
background-size: cover;
background-attachment: fixed;
```

Clipping

```
clip-path: polygon(50% 0%, …);
```

Filters

```
filter: grayscale(70%);
filter: blur(3px);
filter: brightness(0.5);
filter: contrast(200%);
filter: drop-shadow(10px 10px 10px grey);
filter: hue-rotate(90deg);
filter: invert(50%);
filter: saturate(25%);
filter: sepia(50%);
filter: opacity(50%);
filter: grayscale(70%) blur(3px);
```

Supporting High-density Screens

```
<img
   srcset="
      images/meal.jpg     1x,
      images/meal@2x.jpg  2x,
      images/meal@3x.jpg  3x
   "
   src="images/meal.jpg"
/>
```

Use this for fixed-size images. The browser will download the best image based on the pixel ratio of the device.

Resolution Switching

```
<img
   srcset="
      images/meal.jpg     400w,
      images/meal@2x.jpg  800w,
      images/meal@3x.jpg  1200w
   "
   sizes="
      (max-width: 500px) 100vw,
      (max-width: 700px) 50vw,
      33vw
   "
   src="images/meal.jpg"
/>
```

Use this for flexible-size images. The browser will download the best image that requires less resizing based on the pixel ratio and screen resolution of the device.

Supporting Modern Image Formats

```
<picture>
    <source type="image/webp" srcset="…" />
    <source type="image/jpeg" srcset="…" />
    <img src="…" alt="…" />
</picture>
```

Art Direction

```
<picture>
    <source media="(max-width: 500px)" srcset="…" />
    <source media="(min-width: 501px)" srcset="…" />
    <img src="…" alt="…" />
</picture>
```