Images responsives pour écrans haute densité

Face à l'émergence, ces dernières années, des écrans haute densité (retina chez Apple, HDPI pour Android) rendant les pixels pratiquement indécelables à l'œil nu, il est devenu nécessaire d'adapter ses sites ou applications pour gérer convenablement l'affichage des images et éviter des images floues.

Nous ne présenterons sur cette page que quelques solutions basiques pour adapter facilement vos images aux écrans haute densité.

Images SVG :

Priorité aux SVG (scalable Vector Graphics) :

Ce sont des images vectorielles se basant sur une syntaxe XML et qui peuvent être redimensionnées sans perte de qualité (contrairement aux images jpeg, png …).

Ce sont donc des images totalement responsives et peu volumineuses.

Si ce format fait le bonheur des développeurs pour responsiver les images, il est cependant inadapté pour les photos …

Produire un SVG :

Comme le SVG est du XML, il est tout à fait possible de créer un SVG manuellement …

Sinon, il existe de bons logiciels si vous ne voulez pas mettre la main dans le code :

- Inkscape (libre et OpenSource)
- Adobe Illustrator
- SVG Edit
- …

Utiliser un SVG :

- Avec la balise <object> :

  Certainement la manière la plus sûre et la plus usuelle. D'autant qu'il est possible d'intégrer une solution de repli dans le cas où le navigateur ne supporte pas le SVG.

```
<object data=" logo-esup.svg " width="110" height="110" type="image/svg+xml">
  <!-- Solution de repli pour les navigateurs qui ne comprennent pas en charge le format SVG -->
  <img src=" logo-esup.png" width="110" height="110" alt="ESUP Portail" />
</object>
```

- Avec la balise <img> :

  Pour les navigateurs prenant en charge ce type d'images.

```
<img src="logo-esup.svg" width="110" height="110" alt="ESUP Portail" />
```

- Avec la balise <svg> :

  HTML5 introduit une nouvelle balise qui permet d'intégrer directement le contenu du SVG dans une page web

```
<body>
  <svg width="110" height="110">
    <!-- le reste du code SVG ici -->
    <foreignObject width="0" height="0" overflow="hidden">
```

```
            <!-- Solution de repli pour les navigateurs qui ne comprennent pas en charge le format SVG -->
            <img src=" logo-esup.png" width="110" height="110" alt="ESUP Portail" />
        </foreignObject>
    </svg>
</body>
```

- Dans le code CSS :

  En utilisant la propriété background-image :

```
div {
    background-image: url(logo-esup.svg);
}
```

Remarque :

Certaines de ces solutions ne sont pas compatibles tous navigateurs, pour plus de détails vous pouvez consulter le tableau de Can I Use.

Les media queries :

Très utilisés pour le RWD, les media queries sont également pratiques pour déterminer la densité de pixels d'un écran :

```
.icon{
  display: block;
  width: 110px;
  height: 110px;
}
@media only screen and (-webkit-max-device-pixel-ratio: 1.5),
 only screen and (-o-max-device-pixel-ratio: 3/2),
 only screen and (max--moz-device-pixel-ratio: 1.5),
 only screen and (max-device-pixel-ratio: 1.5) {
  .icon{
    background: url(images/ logo-esup.png) no-repeat;
  }
}
@media only screen and (-webkit-min-device-pixel-ratio: 1.5),
 only screen and (-o-min-device-pixel-ratio: 3/2),
 only screen and (min--moz-device-pixel-ratio: 1.5),
 only screen and (min-device-pixel-ratio: 1.5) {
  .icon{
    background-image: url(images/ logo-esup2x.png);
    /* nouvelle image de 220px par 220px */
```

```
    background-size: 110px 110px;
  }
}
```

Images à 200% :

Une des solutions un peu barbare mais efficace consiste à doubler la taille de vos images :

Par ex. si vous voulez afficher une image de taille 110 px par 110 px, vous intégrez cette image avec des tailles doubles et vous diminuez à 50% :

```
<img src="logo-esup.png" width="110" height="110" alt="ESUP Portail" />
< !—l'image source fait en réalité 220px par 220 px -->
```

Ainsi, vous aurez suffisamment de pixels pour l'afficher sur écran haute densité.

Un inconvénient majeur à cette méthode :

On impose le téléchargement d'une image lourde à tous les utilisateurs ; y compris ceux dont les terminaux n'ont pas besoin d'une telle précision. On augmente donc le temps de téléchargement !

L'attribut srcset :

Il permet de définir une image adaptée au terminal en fonction de la taille de l'écran et de se densité de pixels.

Par ex. :

```
<img src="logo-esup.png"
       srcset="logo-esup.png 1x,
       srcset="logo-esup-320.png 320w 1x,
       srcset="logo-esup-640.png 320w 2x"
       width="850" height="475" alt="ESUP Portail">
```

Dans l'exemple ci dessus, l'image chargée par défaut fait 850x475 pixels. Cela permet aux navigateurs, qui ne prennent pas en charge l'attribut srcset, de charger une image de contenu.

De même pour les autres navigateurs qui savent gérer l'attribut mais qui ne trouvent aucune correspondance dans les critères de srcset.

Les alternatives possibles pour chaque image sont :

- 1x : densité de pixel à 1 (est identique à src)
- 320w 1x : propose une image de 320 pixels de large
- 320w 2x : propose une image de 320 pixels de large et ou la densité est inférieure ou égale à 2

L'attribut srcset peut également être couplé avec l'attribut sizes :

Par ex. :

```
<img src="logo-esup.png"
       srcset="logo-esup.png 1x,
       srcset="logo-esup-320.png 320w 1x,
       srcset="logo-esup-640.png 320w 2x"
       width="850" height="475" alt="ESUP Portail"
       sizes="(min-width: 600px) 200px, 50vw">
```

Quand la condition est vraie (min_width: 600px) l'image aura une largeur de 200px , sinon 50% du viewport.

A noter toutefois, la faible compatibilité avec certains navigateurs (ie par ex.) : http://caniuse.com/#feat=srcset