Comprendre le Viewport dans le Web mobile

Le **Viewport** désigne schématiquement la surface de la fenêtre du navigateur. Cependant, la notion de viewport sur un appareil mobile est différente de celle sur un écran de bureau : sur mobile, le navigateur ne dispose pas de "fenêtre" réelle, ni de barres de défilement car tout est prévu pour naviguer de manière tactile.

Comprendre la notion de viewport est absolument indispensable dans un projet d'intégration de site web pour tablettes et smartphones, ou dans un esprit d'adaptation "Responsive Web Design".

Les différentes surfaces d'un mobile

Afin de mieux cerner et exploiter le concept de viewport sur terminaux nomades, il nous faut commencer par maîtriser deux notions de base : la surface réelle et la surface en "pixels CSS" des mobiles.

La surface réelle

C'est le nombre physique de pixels qui composent la matrice de l'écran, telle que le constructeur le décrit dans les caractéristiques, en gros la "résolution" (en vérité le terme juste est "définition").

Par exemple, la *surface réelle* de quelques terminaux Apple :

- 320x480px pour l'iPhone 3
- 640x960px pour l'iPhone 4
- 640x1136px pour l'iPhone 5
- 750x1334 pour l'iPhone 6 (1080x1920 pour l'iPhone 6+)
- 768x1024px pour l'iPad 2
- 1536x2048px pour l'iPad 3, 4, Air

La surface en "pixels CSS"

Egalement appelée `device-width` (`device-height`) ou `screen.width` (`screen.height`) ou encore "ça dépend", il s'agit du nombre de pixels virtuels que le terminal pense avoir et sur lequel il fonde son affichage.
Le hic est que cette surface ne correspond pas toujours à la surface réelle, notamment pour les mobiles dits "retina" ou haute définition. Un "pixel CSS" n'est donc pas égal à un pixel physique.

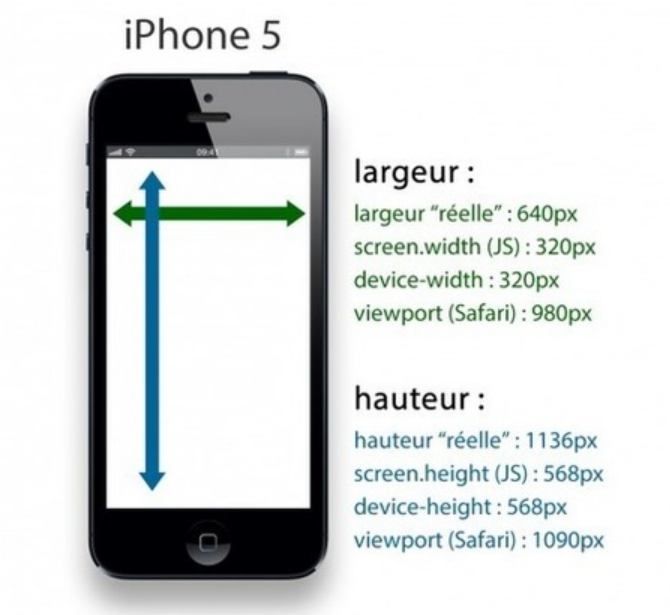
Par exemple, la surface `device-width` et `device-height` en "pixels CSS" (ou "pixels indépendants" DIP) de quelques terminaux

Apple (pour ne citer qu'eux) :

- 320x480px pour l'iPhone 3
- 320x480px pour l'iPhone 4
- 320x568px pour l'iPhone 5
- 375x667 pour l'iPhone 6 (414x736 pour l'iPhone 6+)
- 768x1024px pour l'iPad 2
- 768x1024px pour l'iPad 3

Vous vous doutez bien qu'il en est de même pour un grand nombre d'autres terminaux, toutes marques confondues. Pour avoir un panel de périphériques plus important, je vous invite à découvrir http://mydevice.io/devices/ » http://mydevice.io/devices/ ou encore http://screensiz.es » http://screensiz.es

Attention, les terminaux Apple ont une particularité que n'ont pas les autres mobiles : la valeur de `device-width` est **invariable** en portrait et en paysage. Le `device-width` d'un iPad vaudra toujours 768px quelle que soit l'orientation. Plus de détails dans la partie "Portrait et Paysage" ci-dessous...



Viewport : les mobiles mentent !

Suite à cette introduction aux différentes tailles des mobiles particulièrement déroutante, je vous suggère de continuer à vous accrocher… parce que la notion de viewport est loin d'être une évidence non-plus !

Pour débuter en douceur, apprenez que par défaut la taille du viewport d'un terminal mobile ne correspond ni à la taille de son écran réelle ni celle en "pixels CSS".
Elle est généralement bien supérieure à la surface physique, afin de pouvoir y caler n'importe quelle page web en lui affectant un niveau de (dé)zoom.

Autre surprise, la valeur initiale du viewport ne dépend pas du terminal, comme on pourrait le supposer, mais... du navigateur mobile (et peut parfois même être modifiable par l'utilisateur dans ses réglages).

Voici quelques valeurs par défaut :

- Android 1, 2 et 3 : 800px
- Android 4 : 980px
- Opera mini : 850px
- Opera mobile : 980px
- Safari mobile : 980px
- Internet Explorer mobile : 1024px

Niveau de zoom initial

Compte tenu de ces différentes surfaces, les pages web s'affichent par défaut de manière à ce que toute la surface entre dans celle de l'écran.

> Ce niveau de zoom initial correspond à une simple division mathématique de `device-width` / `viewport`.

Au final, la surface réelle en pixel n'est pas prise en compte dans le calcul du niveau de zoom d'affichage.

Par exemple, Safari mobile sur iPhone 5 va afficher par défaut les pages web dans une fenêtre de 980px de large au sein des 320px de largeur qu'il croit avoir… bien qu'il en ait physiquement 640px. **Le niveau de zoom initial sera de 320 / 980, soit environ 0.326x.**

Illustration : site de alsacreations.com vu par un mobile (par défaut)

La balise <meta> viewport

Fort heureusement, pour s'affranchir de ce zoom intempestif rendant les contenus illisibles, il est possible de modifier et d'imposer la taille de la surface du viewport d'un périphérique mobile. Non pas en CSS comme on pourrait le croire, mais en... HTML, à l'aide d'un élément `<meta>` proposé initialement par Apple.

Les différentes valeurs de cet élément `meta` et de son attribut `content`, offrent la possibilité de fixer la largeur de viewport à la valeur souhaitée, voire de l'adapter automatiquement à la valeur de `device-width` du terminal.

Illustration : page sans meta viewport

Observation : le niveau de (dé)zoom est de 320/980 = 0.3. **Le navigateur mobile considère que sa fenêtre a une largeur de 980px.**

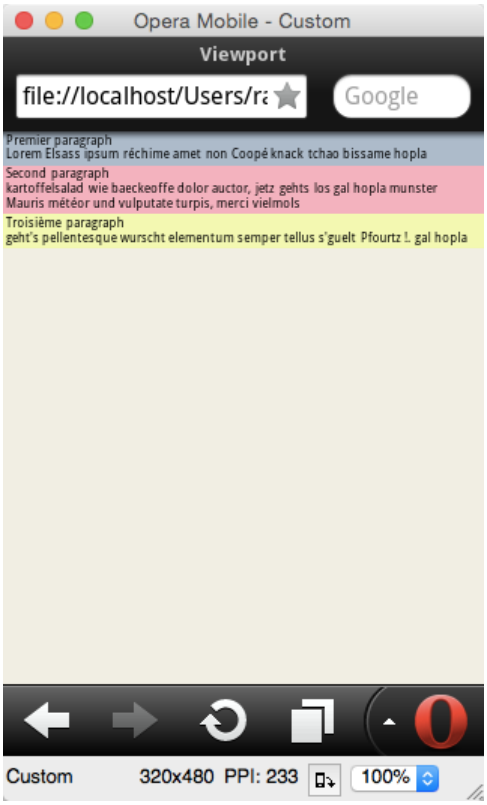
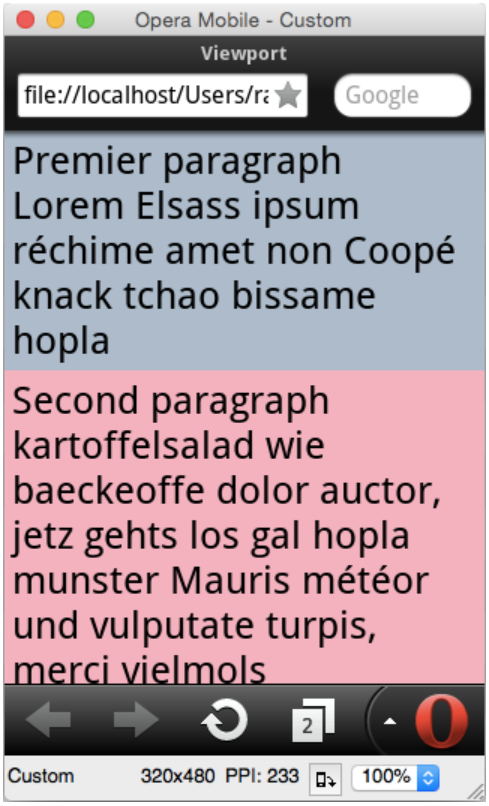


Illustration : page avec une meta viewport

Observation : le niveau de zoom est de 1. **Le navigateur mobile considère que sa fenêtre a une largeur de 320px.**



Insérer une balise <meta> viewport

Afin de forcer le bon niveau de zoom aux navigateurs mobiles, mais aussi pour leur imposer une largeur de viewport qui n'est pas égale à 980px (ou autres valeurs selon le navigateur), il vous sera nécessaire d'inclure une balise `<meta>` "viewport" au sein du `<head>` de votre document HTML.

Code HTML correspondant :

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

Cette balise suffit en général à régler tous les soucis de dézoom.

Autres valeurs de la balise <meta> viewport

La balise `<meta>` autorise d'autres valeurs utilisables dans vos projets :

- `width`
  largeur de fenêtre viewport (par exemple `width="device-width"`)
- `height`
  hauteur de fenêtre viewport (par exemple `height="device-height"`)
- `initial-scale`
  niveau de zoom initial (par exemple `initial-scale="1.0"`)
- `minimum-scale`
  niveau de zoom minimal (par exemple `minimum-scale="0.5"`)
- `maximum-scale`
  niveau de zoom maximal (par exemple `maximum-scale="3.0"`). Attention, la valeur "1.0" interdit le zoom et peut rendre vos pages inaccessibles
- `user-scalable`
  possibilité à l'utilisateur de zoomer (par exemple `user-scalable="yes"`). Attention, la valeur "no" interdit le zoom et peut rendre vos pages inaccessibles
- `target-densitydpi`
  choix de résolution, en dpi, de l'affichage général (spécifique Webkit et semble avoir été abandonné)

Du côté des spécifications : @viewport

En y repensant, j'ai oublié de préciser un détail important en ce qui concerne la balise HTML `<meta> viewport`, mais vous l'aurez peut-être deviné par vous-même : il se trouve que cet élément, inventé par Apple, est bien entendu complètement propriétaire et étranger à toute spécification homologuée.
Fort heureusement, la grande majorité des navigateurs mobiles l'ont adopté en vertu de ses bénéfices pour l'intégrateur.

Sachez que le W3C intègre cette fonctionnalité au sein de son équivalent standard, sous forme d'une règle-at en CSS, ce qui semble ma foi plus logique : `@viewport`.

Cette règle-at est actuellement déjà implémentée sur Opera mobile et IE mobile » http://caniuse.com/#feat=css-deviceadaptation , et se présente sous la forme suivante :

```
@viewport {
   width: device-width; /* largeur du viewport */
   zoom: 1; /* zoom initial à 1.0 (et clin d'oeil aux fans d'IE6/7) */
}
```

Lorsque cette fonctionnalité standard sera implémentée plus massivement (notamment sur Safari mobile et Android), il sera temps de tourner la page et de laisser de côté nos chères balises `<meta> viewport`. Mais c'est encore quelque peu prématuré...

Conclusion et ressources

Au vu de ce (long) article, vous aurez compris que le Web mobile demeure encore parfois difficile à saisir et que bon nombre de ses particularités restent encore obscures.

J'espère que toutes ces explications et exemples visuels vous auront permis de mieux comprendre le concept général du viewport sur les mobiles et de mettre en oeuvre vos propres solutions pour le dompter.

Quelques ressources très utiles pour finir :

- Apple : Using the Viewport (anglais) » http://developer.apple.com/library/IOS/#documentation/AppleApplications/Reference/SafariWebContent/UsingtheViewport/UsingtheViewport.html
- Android : targeting screends from web apps (anglais) » http://developer.android.com/guide/webapps/targeting.html
- First : understand your screen (anglais) » http://tripleodeon.com/2011/12/first-understand-your-screen/
- mydevice.io » http:/mydevice.io : affichage des infos utiles de votre navigateur mobile
- Introduction au Web mobile » /article/lire/1464-web-mobile-introduction-et-glossaire.html
- Une feuille de styles de base pour le Web mobile » /astuce/lire/1177-une-feuille-de-styles-de-base-pour-le-web-mobile.html