

Qu'est-ce que le Responsive Design ?



Le Responsive Web Design (**RWD**) est une conception permettant à un site web d'adapter son affichage à n'importe quels types d'appareils et d'écrans.

Pourquoi le responsive design ?

L'objectif des sites prévoyant le responsive design est d'offrir une navigation adaptée et la meilleure expérience utilisateur possible, quel que soit le terminal utilisé par l'internaute.



Depuis 2010 cette technique est régulièrement utilisée.

Comment faire pour adapter son site à chaque écran ?

Une version de site par écran ? Trop long... Nous allons donc créer une interface de site web qui pourrait s'auto-adapter à chaque écran

Cas d'étude : 2 zones à afficher sur une page web.

- Sur écran d'ordinateur, surface large, zone côte à côte.
- Sur smartphone, la surface réduite, zone l'une en dessous de l'autre.

Très grand écran (ordinateur de bureau)	Très petit écran (mobile - smartphone)
	

Les principaux terminaux

Les principaux terminaux et tailles :

Écran	Écran minimum	Écran réduit	Écran moyen	Grand Écran
Illustration				
Type	SmartPhone	Tablette	Ordinateur Portable	Ordinateur de bureau
Résolution d'écran taille en largeur (width)	< 768 px	≥ 768 px < 992 px	≥ 992 px < 1200 px	≥ 1200 px

Nous pouvons également citer les tv, liseuse, watch, console, etc.

Les tailles

Des cas intermédiaires existent, voici les tailles généralement utilisées :

Appareils	Largeurs (en pixels)	Condition généralement utilisée
Smartphones	320 à 480	(max-width: 480px)
De smartphones à tablettes	481 à 767	(max-width: 767px)
Tablettes à petits écrans	768 à 979	(min-width: 768px) and (max-width: 979px)
Ordinateurs	980 à 1199	-
Écrans larges	1200 et plus	(min-width: 1200px)

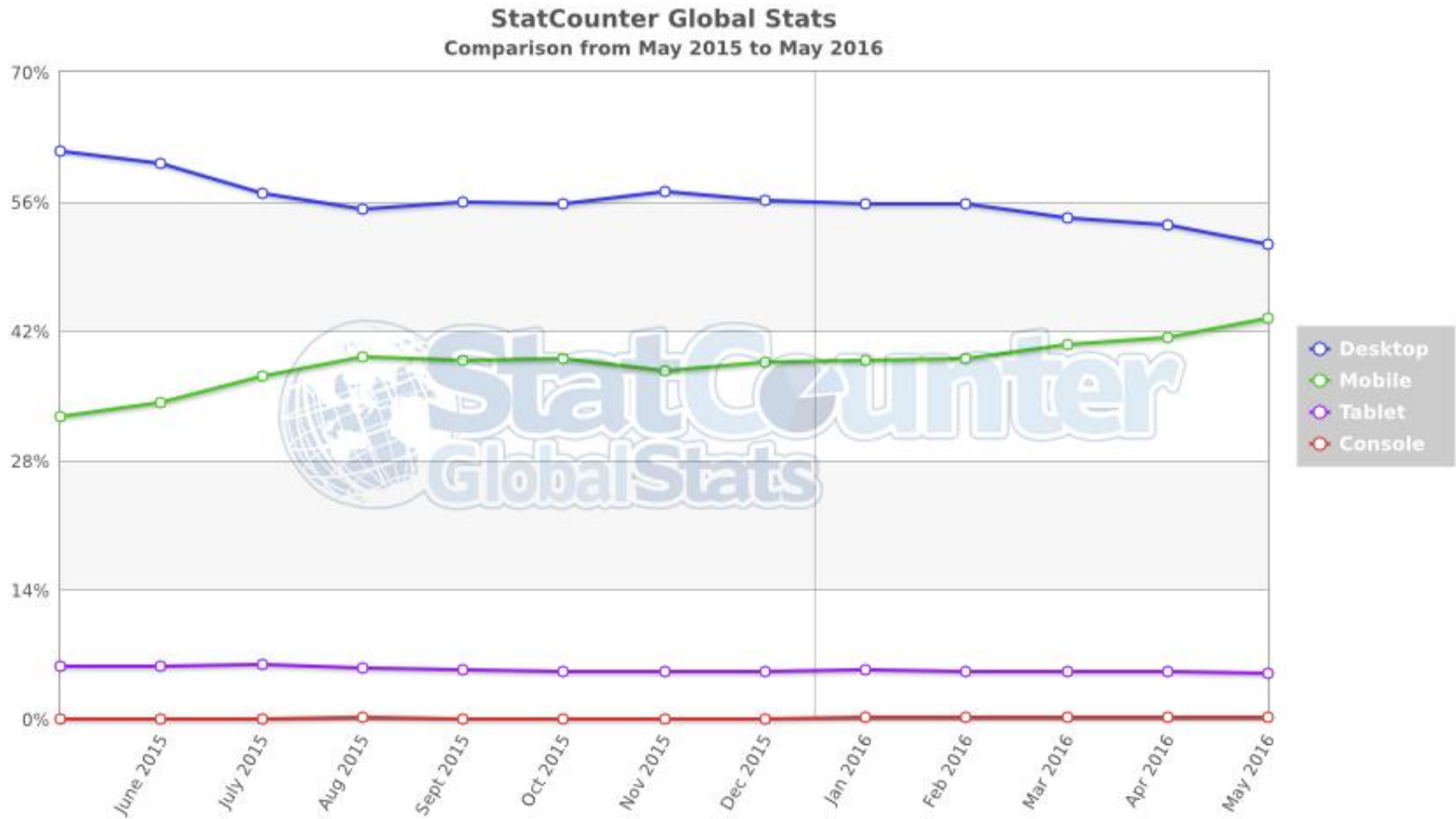
Impact sur le web

Quelques informations autour des terminaux :

- Été 2015, **Google** annonce que les recherches sur smartphones ont dépassé celles effectuées sur PC.
- **Google** pénalise les sites ne prévoyant pas leur affichage sur les différents terminaux de ses internautes.
- Chez **Facebook**, les nouveaux terminaux constituent désormais 76% des recettes publicitaires
- **MicroSoft** a lancé le projet Continuum en 2014
- Les chiffres du **m-commerce** explosent et une stratégie adaptée à ce canal permet un meilleur ROI.

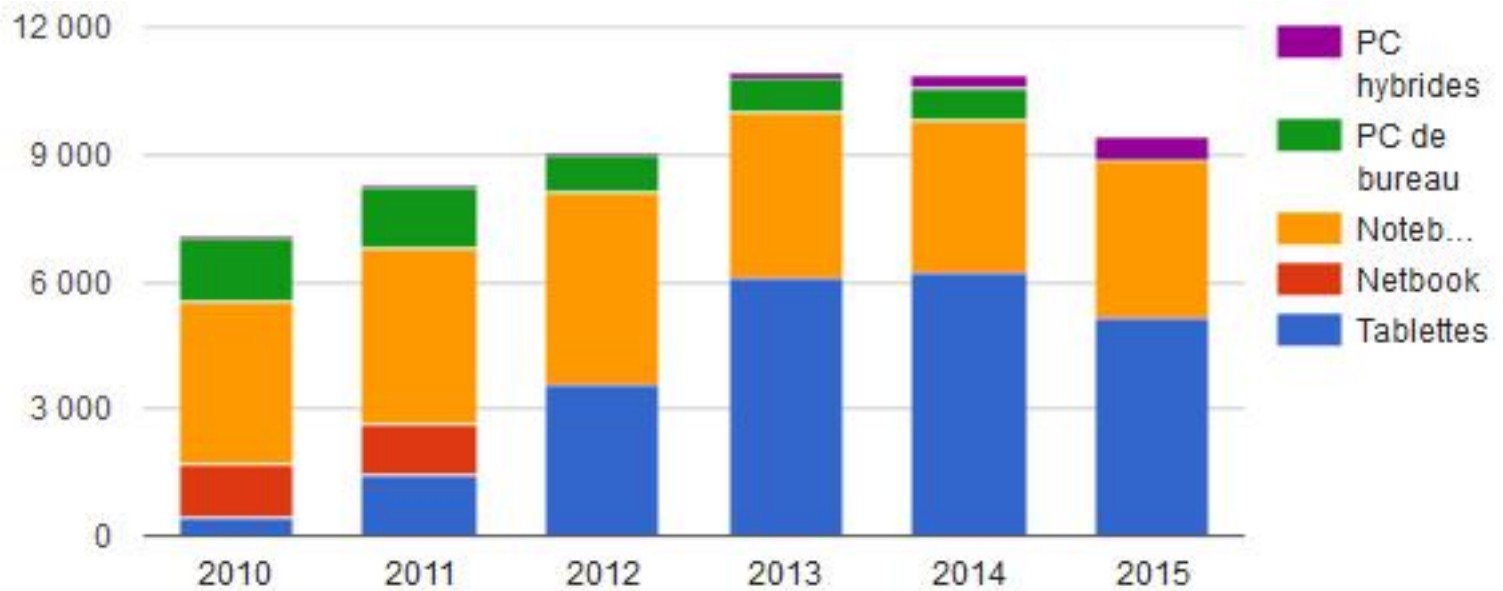


Les parts de marché



Source : StatCounter

Les ventes



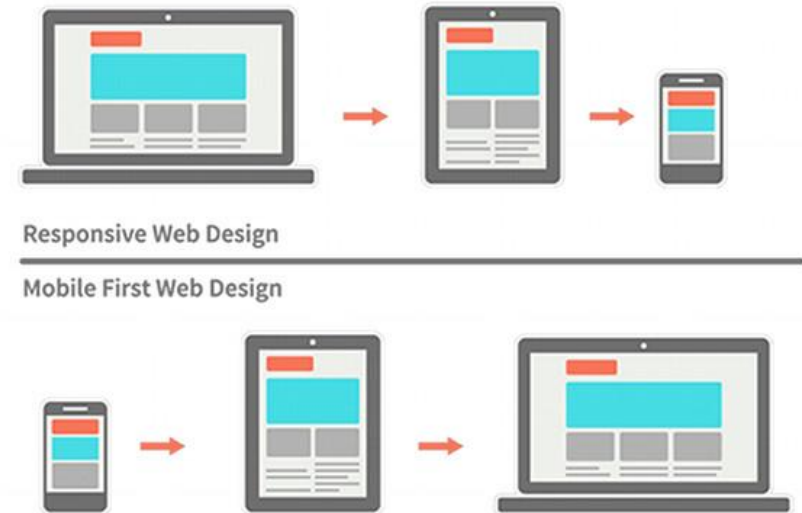
Source : ZDNET

Mobile First

- Concept de Ethan Marcotte, **Mobile First**.

Privilégier la construction d'une interface ergonomique pour **terminal mobile** avant d'établir l'ergonomie pour un terminal « desktop » (bureau).

Cela permet à l'ensemble des acteurs (designers, ergonomes, commanditaires) de **rester centrés sur les premiers besoins des utilisateurs** et ainsi distinguer les informations importantes des informations superflues (*Graceful Degradation*).



Mobile First ou non, il est indispensable de **penser à la version responsive dès le début d'un projet** et de l'adapter en cours de route. Évitant ainsi d'y réfléchir une fois que le site est terminé.

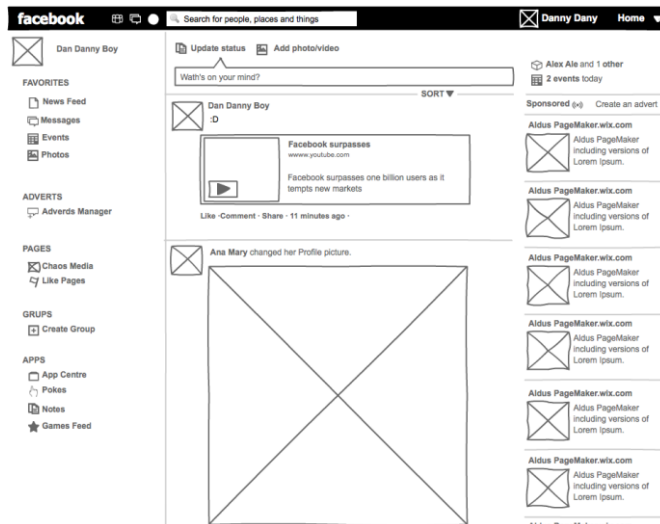
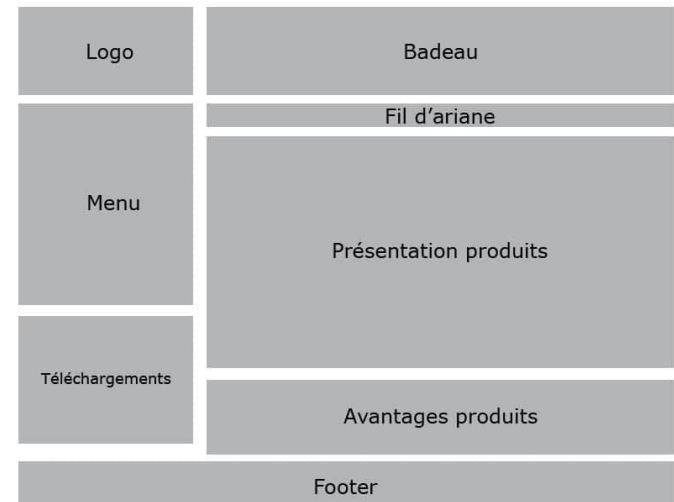
Composants Graphiques

Zoning = Agencement des blocs

WireFrame = Schéma fonctionnel

Mockup = Interface web

Maquette = Graphisme du site



Balsamiq <https://www.balsamiq.com/>
permet d'élaborer des wireframe et
mockup.

UX DESIGN : Expérience Utilisateur

Qu'est-ce que l'UX DESIGN ?

L'UX DESIGN c'est proposer à l'internaute une **navigation simple et claire** en fonction de ses habitudes.

Pour cela il faut prendre en compte la **psychologie de l'être humain** en estimant les zones chaudes et froides de la page web (*l'endroit où se pose l'œil humain en priorité*).



L'expérience Utilisateur

Pour renforcer l'expérience de l'utilisateur nous devons **anticiper ses attentes et ses usages**.

La page web doit être adaptée à l'utilisateur et non pas l'inverse.



UX DESIGN en complément de l'ergonomie

L'ergonomie nous permet de prévoir un agencement pertinent. L'UX design va plus loin :

- **Utile** : avec seulement les fonctionnalités importantes.
- **Utilisable** : l'utilisation doit être facilitée.
- **Désirable** : le visuel (émotionnel) a beaucoup d'importance chez la plupart des êtres humains.
- **Navigable** : il n'est pas envisageable de perdre du temps à chercher des informations pour se déplacer dans le site.
- **Accessible** : prévu pour les personnes ayant des handicaps.
- **Crédible** : un design UX génère une opinion positive chez l'internaute.



L'UX s'est imposée depuis le début des années 2010 comme un atout majeur du marketing et est au cœur des stratégies d'entreprise. Exemple => <https://www.airbnb.fr/>

Type : Fixe, Fluide ou Adaptatif

➤ Fixe

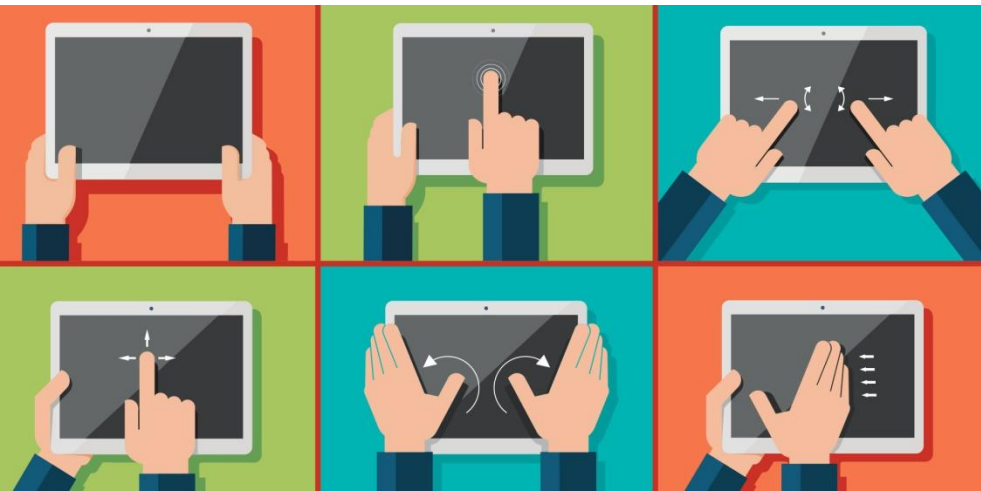
Le site est créé dans des mesures **fixes** souvent exprimées en pixels.
La taille ne change pas selon l'écran.

➤ Fluide (ou liquid)

Le site est créé dans des mesures **variables** souvent exprimées en pourcentage. Unité de mesure privilégiée : pourcentage, em, vh, vw, etc.

➤ Adaptatif

Le site est créé dans des mesures fixes mais **diffère selon la taille d'écran**.
Ces conditions sont exploitées grâce aux **médias-queries**.



Un design responsive, c'est généralement une base fluide associée à des cas adaptatifs.

Options : Responsive, Version mobile, Application mobile

➤ Responsive

Avantages

Une seule **version adaptée** pour chaque écran.

Une maintenance plus simple (un seul site à mettre à jour).

Inconvénients

De nombreux tests variés tout au long du projet

➤ Version Mobile

Avantages

Forte personnalisation : Un site dédié au mobile (*généralement : m.site.com*) permet de cibler précisément la structure, le contenu et les fonctionnalités pour le mobinaute.

Inconvénients

Le contenu est dupliqué « duplicate content » (*deuxième url*) et cela oblige donc à réécrire un deuxième site.

➤ Application Mobile

Avantages

Prise en charge facilitée de **fonctionnalités natives** (touch, notifications, GPS, etc.)

Inconvénients

Une action de l'utilisateur est nécessaire pour installer l'application sur son smartphone.



Détection du terminal

➤ Détection côté client

Avec l'utilisation de la technologie JavaScript, la détection du terminal côté client permet d'adapter la page web à la taille et aux caractéristiques du terminal utilisé.

➤ Détection côté serveur

La détection de terminal côté serveur permet de sélectionner dans la base de données uniquement les ressources dont l'internaute a besoin (*HTML, CSS, images*), au lieu de charger la totalité des données pour ensuite les masquer/afficher.

➤ Détection des terminaux

WURFL, DeviceAtlas, DetectRight et UADetector proposent une base de données à jour comprenant les différents terminaux mobiles existants, pour détecter et récupérer les caractéristiques d'un terminal plus facilement (*à partir de la requête HTTP*).

➤ Détection des fonctionnalités

Modernizr est une bibliothèque développée en JavaScript détectant l'implémentation native par le navigateur de différentes fonctionnalités récentes.

L'organisation **W3C** travaille sur une évolution permettant de prendre en compte des informations sur la qualité de la connexion internet (côté client).



Système de grille

Qu'est-ce qu'une grille ?

Une grille représente un « quadrillage » de la page web.

Concevoir son intégration avec une grille aide à structurer sa page web en facilitant l'écriture du code css (*proportion équivalente d'une page à l'autre*).

Cela permet également aux webdesigners de prévoir un graphisme en adéquation avec la future intégration et ainsi de prendre en compte les contraintes et problématiques de conception.

Hauteur

La hauteur d'un site n'est pas fixe et dépend généralement de son contenu.

Largeur

La largeur d'un site est souvent de 960 px car ce nombre est divisible par 1 2 3 4 5 6 8 10 12 15 16 20 24 30 32 40 48 60 64 80 96 120 160 192 240 320 480 960. Cela offre beaucoup de possibilités de mises en pages. Pour autant cela n'est pas une obligation absolue.

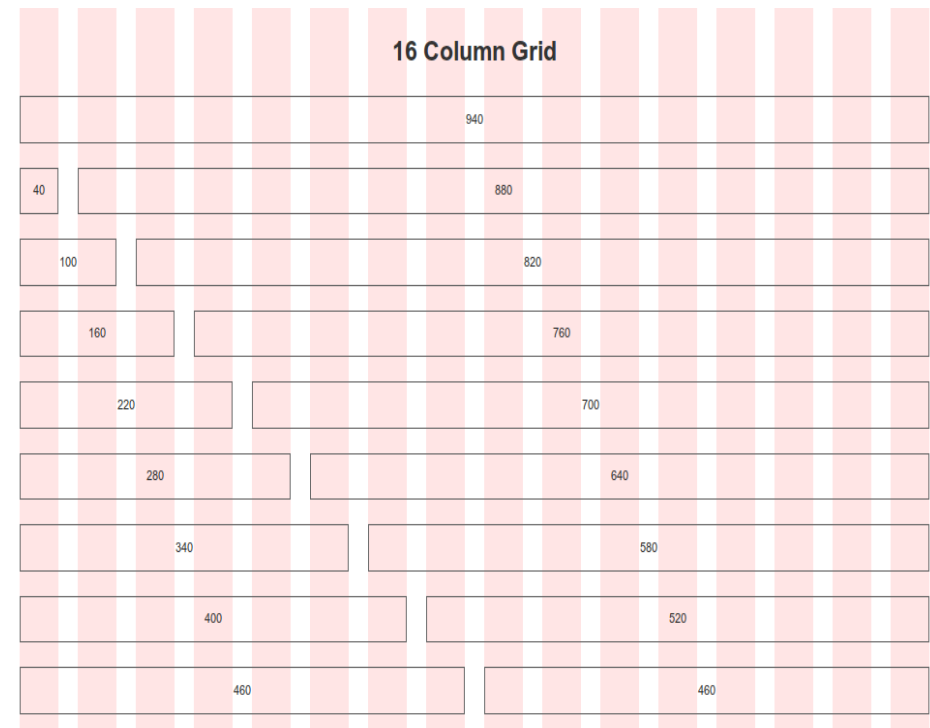
Nombre de colonnes

Le nombre de colonnes d'une grille est souvent de 12, 16 ou encore 24.

Dimensions

16 x 50px = 800px (taille des colonnes)

16 x 10px = 160 px (espacement entre les colonnes)



Total = 960 px

FrameWork

Qu'est-ce qu'un framework ?

Un framework est avant tout un cadre de travail permettant de faire profiter aux développeurs d'une arborescence et d'une architecture communes.

Quel est l'intérêt d'utiliser un framework CSS ?

L'utilisation d'un framework permet de faciliter l'écriture de la mise en forme CSS d'un site web.

Comment ça fonctionne ?

Le code CSS est déjà établi et nous utilisons les classes pré-existantes pour assurer la conception de notre page web.

Quels sont les frameworks CSS à privilégier ?

Les plus connus/utilisés : Bootstrap, 960Grid, Foundation, etc.



FrameWork : Avantages

Gain de temps

Le développement front d'un site sera plus rapide car le code css est déjà prêt.

Conception

Le système de grille facilite l'agencement, le positionnement et l'alignement des blocs.

Responsive

Etudié pour l'adaptabilité, la base (système de grille) est responsive, ce qui est indispensable de nos jours avec la multitude de terminaux.

Maintenance

Le code étant commun à tous les développeurs , cela facilite la maintenance et la reprise de projet.

Compatibilité

Le code est pensé pour être compatible sur tous les navigateurs.

Graphisme

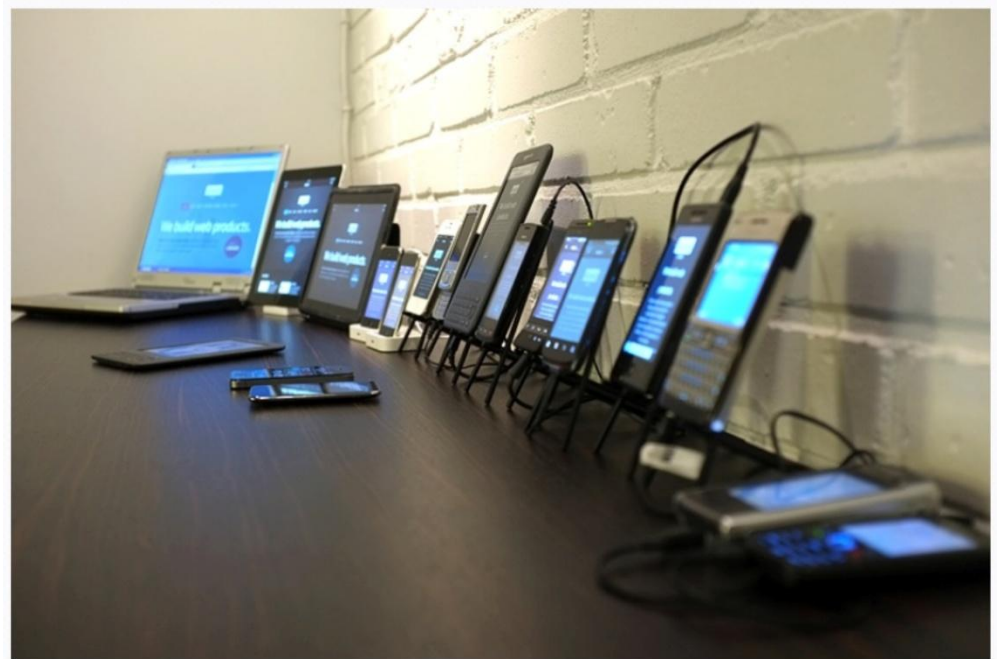
Il existe plusieurs thèmes préfaits de haute qualité pour obtenir une interface fonctionnelle et intuitive rapidement.

Emulateurs






Les **émulateurs** permettent d'effectuer des **tests** sur différents **terminaux** :

- <http://www.responsinator.com/>
- <http://beta.screenqueri.es/>
- <http://quirktools.com/screenfly>

*Dans la mesure du possible, il est préférable
de faire des tests sur les terminaux
originels.*



Compatibilité navigateurs

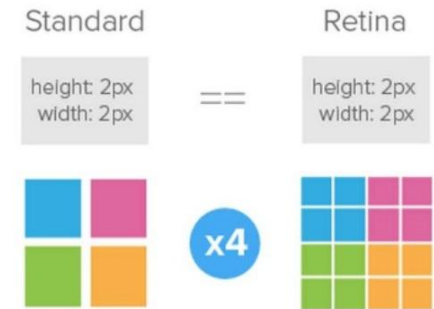
Navigateur	Google Chrome	FireFox	Internet Explorer / Edge	Safari	Opera
2016	68 %	19.1 %	6.3 %	3.7 %	1.5 %
2015	63.9 %	21.6 %	8 %	3.8 %	1.5 %
2014	55.7 %	26.9 %	10.2 %	3.9 %	1.8 %
2013	48.4 %	30.2 %	14.3 %	4.2 %	1.9 %
2012	35.3 %	37.2 %	20.1 %	4.3 %	2.4 %
2011	23.8 %	42.8 %	26.6 %	4 %	2.5 %
2010	10.8 %	46.3 %	36.2 %	3.7 %	2.2 %
2009	3.9 %	45.5 %	44.8 %	3 %	2.3 %
2008	Inexistant	36.4 %	54.7 %	1.9 %	1.4 %
2007	Inexistant	31 %	58.6 %	1.7 %	1.5 %
2006	Inexistant	25 %	66 %	Inexistant	1.6 %
-					

ViewPort

Le **Viewport** désigne schématiquement la surface de la fenêtre du navigateur.

La Surface en pixel physique

Il s'agit du nombre de pixels physiques qui représentent la résolution d'écran.



La surface en pixels css

Il s'agit du nombre de pixels virtuels (css) qui représentent la taille totale de la page web.

Surface des différents terminaux

<http://mydevice.io/devices/> (1 pixel physique n'est pas égal à 1 pixel virtuel).

L'affichage

Pour éviter que le navigateur d'un smartphone se base sur sa résolution réelle et effectue un zoom/dézoom automatique, nous utiliserons la balise meta/viewport.

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

Performances

Chaque seconde compte

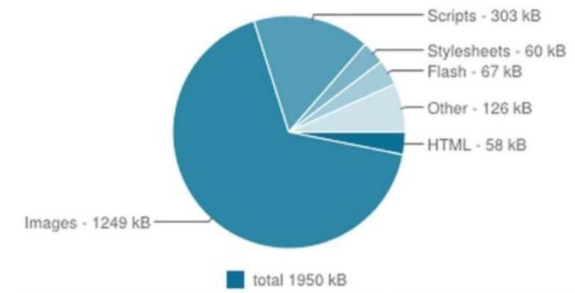
¼ des internautes quittent un site qui met plus de 4 secondes à charger.

Analysons le poids d'une page

<https://developers.google.com/speed/pagespeed/insights/>

<https://www.dareboost.com>

<http://www.webpagetest.org>



Comment optimiser les performances ?

- ✓ Les **images** pèsent souvent plus lourd, il est indispensable de charger l'image aux dimensions réelles d'affichage.
- ✓ Eliminer le code css et javascript qui détermine l'affichage en dessous de la **ligne de flottaison**.
- ✓ Appel des scripts en bas de page (**easy loading**), après que l'affichage soit rendu au navigateur.
- ✓ Réduire le code via la **minification**.
- ✓ Exploiter la **mise en cache** du navigateur.
- ✓ **Réduire les dépendances** aux scripts extérieurs (exemple bouton réseaux sociaux, etc.).
- ✓ **Détecter le terminal** côté serveur pour charger uniquement les ressources liées au besoin de l'internaute (*éviter de tout charger pour faire des display: none;*).