COURS 8

Introduction au responsive

Concevoir pour toutes les contraintes d'écran

582-211 MO Web 2 · H2026 · Marie-Michelle Ouellet

DÉMO: Qu'est-ce qui se passe quand on rétrécit la largeur de la fenêtre ?

Le texte déborde

→ Conséquence visible à l'écran

Les colonnes s'écrasent

→ Conséquence visible à l'écran

Le site devient inutilisable

→ Conséquence visible à l'écran

*Discussion : À quel moment le site devient-il vraiment **inutilisable** pour toi ?*

Le *responsive* : s'adapter à la contrainte

Définition

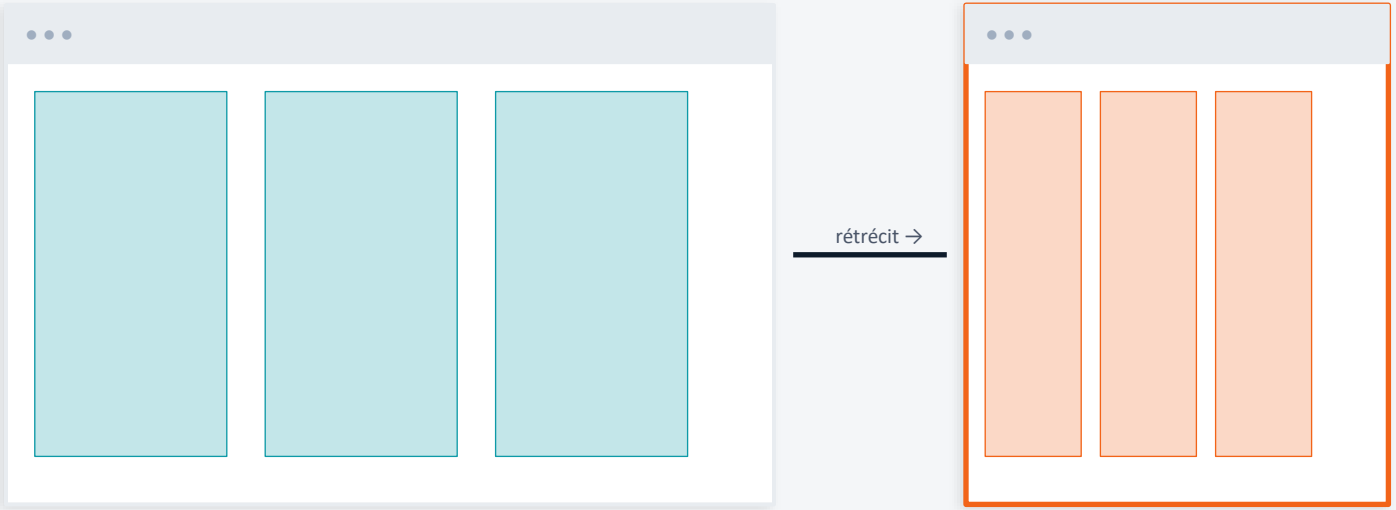
Un design *responsive* **adapte la mise en page** à la **contrainte d'affichage** courante, peu importe l'appareil.

Ce que ce n'est PAS

- ✗ Réparer un site cassé
- ✗ Cibler des appareils spécifiques
- ✗ Une liste de *breakpoints* fixes à mémoriser

"Le responsive commence quand ton contenu commence à souffrir visuellement."

Un *breakpoint* = là où ton contenu souffre



✓ Confortable

✗ Souffrance visuelle ← ici le breakpoint

Règle : Ajoute un breakpoint quand ton contenu te le demande — jamais en ciblant un appareil spécifique.

La syntaxe d'une *media query*

```
@media (min-width: 768px) {
  .conteneur {
    flex-direction: row;
  }
}
```

@media → Mot-clé — démarre la règle conditionnelle

(min-width: 768px) → Condition — s'applique SI la fenêtre est ≥ 768px

{ ... } → Règles CSS qui s'appliquent seulement si la condition est vraie

min-width vs max-width

```
/* Mobile-first (recommandé) */
/* Styles de base = mobile */

@media (min-width: 768px) {
  /* ajoute pour le grand */
}
```

✓ **Mobile-first**
On part du petit, on ajoute

```
/* Desktop-first */
/* Styles de base = desktop */

@media (max-width: 767px) {
  /* retire pour le petit */
}
```

△ Desktop-first
On part du grand, on enlève

Mobile-first : partir de la contrainte

Mobile-first

Partir du minimum essentiel et ajouter de la complexité au fur et à mesure que l'espace augmente.

```
.nav { flex-direction: column; } /* mobile */

@media (min-width: 768px) {
  .nav { flex-direction: row; }
}
```

On ajoute ↑

Desktop-first

Partir de la version riche et retirer des éléments au fur et à mesure que l'espace diminue.

```
.nav { flex-direction: row; } /* desktop */

@media (max-width: 767px) {
  .nav { flex-direction: column; }
}
```

On retire ↑

Analyser avant de coder

01	Quels éléments changent entre mobile, tablette et desktop ?
02	Combien de *breakpoints* peux-tu deviner ?
03	Quelle version a été pensée en premier ?
04	Quelle propriété CSS est impliquée dans chaque changement ?

Travail en tandem → mise en commun en classe

Première *media query* : à toi de jouer

Mission : Une carte en colonne (mobile) → en rangée (desktop)

```
/* À ajouter dans ton CSS */
@media (min-width: ???) {
  .carte {
    flex-direction: ???;
  }

  .carte__image {
    width: ???;
    flex-shrink: 0;
  }
}
```

1. Créer **exerc-mediaqueries/index.html** avec le code fourni sur Timdoc
2. Ajouter la balise meta viewport (sinon rien ne marche)
3. Choisir un *breakpoint* en observant le contenu
4. Tester : rétrécir et agrandir la fenêtre
5. Vérifier : aucun overflow horizontal (donc pas de scrollbar horizontale à la page)

⚠ **Sans <meta name="viewport"> → les media queries ne fonctionnent pas sur mobile.**

Qu'est-ce qui a résisté ?

Le viewport meta

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

min-width vs max-width

mobile-first = min-width / desktop-first = max-width

flex-shrink: 0

Sans ça, les éléments flex se compriment même si tu fixes une largeur

Trois niveaux — Avance à ton rythme

Niveau 1

Navigation repliable

Horizontal sur desktop → vertical sur mobile.
Un seul breakpoint.

Niveau 2

Grille de cartes

1 → 2 → 3 colonnes selon l'espace.
Deux breakpoints.

Niveau 3

Composant héros

Réorganisation complète + typo fluide.
Tu écris tout le CSS.

Instructions complètes sur Timdoc · Ce qui n'est pas terminé en classe = devoir pour le cours 9

Responsive ≠ Accessible (mais c'est un début…)

Cibles tactiles ≥ 44px

Les doigts ne sont pas des curseurs.
Un lien de 20px de haut est inutilisable sur mobile.

Texte lisible sans zoom

Corps du texte : 16px minimum.
Si l'utilisateur doit zoomer, c'est ton problème.

Pas de scroll horizontal

C'est la signature d'un site non responsive.
Teste à toutes les tailles.

Espacements adaptés

Les marges et paddings pensés pour desktop sont souvent trop serrés sur mobile.

On approfondit l'accessibilité au cours 11. Aujourd'hui : on pose les bases.

Ce que vous savez maintenant faire

- ✓ Choisir un breakpoint en observant le contenu
- ✓ Écrire une media query avec la bonne syntaxe
- ✓ Appliquer l'approche mobile-first (min-width)
- ✓ Inclure le viewport meta tag (toujours)
- ✓ Tester la convivialité mobile de base

Ajoute un breakpoint quand ton contenu te le demande.

Prochain cours →

Responsive avancé

- Media queries avancées
- orientation, prefers-color-scheme
- Stratégies de breakpoints

Introduction à Grid

- CSS Grid bidimensionnel
- Quand Grid > Flexbox