# Récap Collectif

## Historique et culture générale

- XHTML
    - Notion de sémantique
    - Séparation forme et fond (CSS/ HTML)
- HTML 5
    - Hérite de XHTML
    - Ajoute des balises de sectioning
        - header, footer, aside, section,...
- W3C : consortium
    - Responsable de la cohérence mondiale du web
    - Ils proposent des recommandations
    - Structure à but non lucratif
        - composée de dév logiciel, web, navigateurs, …

## Éléments significatifs des interfaces web

- Layout utile
    - Empecher sur les très grands écrans de perdre en lisibilité
    - Indispensable pour textes
    - Négociables ou blocs de contenus
- Elements sticky
    - sortent du flux et restent fixe sur la zone écran indépendamment du scroll
    - palette outils / navigation / menu principal
        - chat bot
        -
    - utile pour pages longues
- Envisager le réagencement responsive des contenus
    - ex footer boncoin
    - ex favoris coup de coeurs boncoin
- Approche modulaire responsive
    - On peut créer des variations graphiques entre smartphone et desktop
    - Mais garder un fil directeur graphique (cohérence)
- Ligne de flottaison
    - Contenu important doit être au dessus
    - Gérer la surface d’écran (lien avec le sticky) viewport

# VS code et mise en place des fichiers

## Raccourcis et assistances visuelles

- alt+Z  Activer retour à la ligne auto
- Fractionnement des fenêtres en glissé déposé
- ! active la structure HTML de base
    - meta desc (seo)
    - lang fr
    - écrire le title (seo)
- alt+shift F  indentation auto
- alt + clic  duplique le curseur (fonctionne aussi avec sélection)
- ctrl + v (sans sélection) copie colle la ligne entière
- alt + touches directionnelles (déplace la ligne ou la sélection)
- lorem25 (affiche 25 mots)
- ctrl + :  active / désactive le commentaire
- tab et maj + tab  indentation de la sélection

### Syntaxe HTML~#

<nom_de_balise  nom_attribut = " valeur_attribut"  nom_attribut2  =  "valeur">  Contenu pris compte </nom_de_balise>

### Emmet

- >  enfant de premier niveau
- +  elément suivant
- * permet de multiplier le nombre d'éléments générées
- {}  permet de définir le contenu de la balise
- []  écrire les attributs/valeur dans la balise
- () permet d'isoler un ensemble de balises (pour les répétitions)
- .  ajoute une classe a la balise
- #  ajoute un id

### Ciblage relatif

- On part du fichier dans lequel on écrit (css ou html) et on mentionne chaque dossier qu'on traverse jusqu'au fichier linké
- écrire un / entre chaque dossier
- Pour remonter d'un dossier parent ../
    - autant de fois que de dossiers à remonter

## Balises HTML 5

### #Sectionning

- header / footer
    - peuvent être utilisés pour la page ou des sections, aside et autres contenus longs

- main
    - 1 seule fois dans la page
- section
    - une partie d’un ensemble plus grand
- article
    - autonome et publiable sur un réseau social
- aside
    - en rapport non direct avec le contenu
- nav
    - élément de navigation

### Medias

- picture
    - source
    - img
- figure
    - img
    - figcaption
- video
- audio
- canvas

## Méthodologie pré intégration HTML

- Dessiner un schémas du DOM

# CSS

## Portée du CSS

- local
    - Ne s’applique qu'à la page en cours
    - <style> /*CSS applicable à la page*/</style>
- Distant (feuille de style)
    - Lier le fichier CSS
    - s’applique à toutes les pages html liées au fichier CSS

### Syntaxe

```
selecteur {
     propriete  :  valeur  ;
     propriete  :  valeur ;
}
```

# Familles de propriétés CSS

### Texte

- color
- text-transform
- font-size
- font-family
    - Fait référence à plusieurs typos (améliorer la compatibilité)
- text-shadow
- text-align
- …

### Background

- color
- size
- repeat
- image

### Boîte

- Dimensions
    - height / max-height / min-height
    - width /….
- Margin (marges extérieures)
- Padding (marges intérieures)
- border-radius
- box-shadow

### Spécial

- float (sortie du flux)
    - overflow:hidden sur parent
    - clear:both sur les suivants

# Fonctionnement du modèle de boîte

- margin (espace extérieur, espace les blocs les uns des autres)
- padding (espace intérieur,  espace le contenu d'un boîte HTML de sa bordure)
- dimensions (largeur / hauteur)
    - penser à utiliser les tailles avec limite quand c'est nécessaire
    - min-width / max-width / min-height / max-height

- Un espace intérieur de bloc peut être
    - le padding du bloc
    - le margin d'un de ses enfants
    - Toujours l'utiliser l'inspecteur pour savoir s'il s'agit de l'un ou de l'autre
- Les margins des enfants s'appuient sur les paddings des parents
    - quitte a agrandir la boîte
- Arrières plans des boîtes
    - recouvre la boîte jusqu'à la bordure de la boîte
    - Les arrières plans recouvrent les paddings de leur boites
        - ex utiliser image arr plan qui ne recouvre pas le texte

## Cas particuliers

### Fusion des marges

Fusion bloc parent enfant

- Lorsque des descendants partagent le même bord (haut ou bas) il y a fusion des marges le bord en commun se superpose exactement
- La plus grande marge s'applique une seule fois au parent

Fusion blocs suivants

- lorsque 2 éléments se suivent, leur marge en commun fusionnent
- C'est la marge la plus grande qui s'applique 1 seule fois entre les 2

### Box-sizing

- Par défaut la taille résultante d'une boîte est la somme de:
    - width + padding + border
    - ça pose un problème parce que les boites ont une dimension qui peut être différente du width
    - Box-sizing place les padding et les border dans le width -> le width sera toujours la réelle dimension de la boîte

### Inline vs Bloc

Bloc

- placement en colonne (les uns en dessous des autres)
- Respecte le modèle de boîte
- 100% de l'espace disponible

Inline

- placement en ligne (côte à côte)
- Respecte mal
- Se réduit à la taille du contenu

# Tableaux HTML

Un tableau est constitué d'autant de lignes (tr) qu'on veut

Chaque ligne doit contenir strictement le même nombre de cellules (td ou th)

## Balises HTML

- Table
- Caption (titre du tableau)
    - thead (sections de tableaux optionnelles)
    - tbody
    - tfooter
- tr (lignes)
- td (cellules)
- th (cellules d'entête)

## Attributs cellules

- colspan (fusion de cellules horizontales)
- rowspan (fusion de cellules verticales)

# Productivité

2 écoles OOCSS et BEM nous allons faire du BEM sauce Alsacréation

Plongée au coeur de l'OOCSS — @nicoespeon's blog

Key concepts / Methodology / BEM

Bonnes pratiques en CSS : BEM et OOCSS - Alsacreations

## Origine

Nomenclature basée sur les classes HTML

- Faciliter le travail à plusieurs

## Fonctionnement

Différencier les éléments HTML en fonction de leur

- réutilisabilité
- de leur hiérarchie (parent / enfant)

### 3 objets du BEM

- Bloc / Element / modifier
- Composant / Descendant / modifier

Composant

- Pour regrouper des éléments HTML ayant la même fonction
- Pour séparer la mise en forme d'un bloc
    - et sa disposition dans l'interface

Descendants

- Pour sélectionner les différents éléments à l'intérieur d'un composant
- Si un descendant a intérêt à devenir composant, c'est possible

Modifier

- Lorsqu'un groupe d'élément possède une apparence semblable, le modifier permet de faire des ajustements sur certains éléments individuels
    - Propriétés de mise en forme commune au niveau du descendants
    - Les distinctions graphiques seules sont dans le modifier

## Avantages

- Réutilisabilité des composants
- Travail à plusieurs
- Optimisation du code (performances)
    - Spécificité basse (problèmes de priorités de sélecteurs)
    - Pas de ciblage en cascade => plus rapide

# SAAS

## Préprocessseur

- Extension vs code (live sass compiler)
- Watching de saas

### Principe#

- On écrit en sass dans un fichier .scss
- a chaque enregistrement
- Sass génère le fichier .css

### Caractéristiques

Les variables

- contiennent des valeurs de propriété CSS
    - $maVariable : 2px 3px 10px red
    - tous types

Les partiels

- permet de scinder le css en plusieurs partie s pour l'organisation personnelle
- Tout en préservant la performance avec un CSS généré unique*
- Syntaxe
    - _nomDuPartiel.scss
    - @import ' nomDuPartiel';

Ecriture en cascade{

- Permet d'écrire une règle dans une autre avec indentation du CSS
- L'utilisation de & permet de concaténer les différents sélecteurs imbriqués

Les Mixins

- Fragment de css nommé, réutilisable

# Responsive web design et les media queries

## Méthodologie

- Mobile first
    - On commence toujours le CSS mobile en premier
    - Pas de de media query
    - On l'appell**e CSS Globa**l
    - Il débute le fichier CSS
    - Attention: Penser à dimensionner le **navigateur sur une largeur mobile**
- Amélioration progressive
    - Privilégier des Media queries en min-width seules
        - plutôt que en fourchettes
- Les points de ruptures successifs doivent être écrits chronologiquement

## Syntaxe

```
    @media condition1 AND condition2{
         /*Tout le CSS applicable uniquement si les conditions sont résolues*/
    }
```

# Rappel methodo d'intégration globale

1. Partir d'un mockup en image
    a. décliné en diverses versions de périphériques
    b. minima : mobile et desktop
2. Représentation schématique du DOM
    a. Attention se baser sur la version la plus large
    b. Mentionner: le nom des balises
    c. Affiche l'imbrication à l'aide de couleurs
3. Préparer dossier de travail dans vsCode
    a. structurer le dossier racine
        i. dossier : css
        ii. dossier : js
        iii. dossier : img
        iv. fichier index.html
    b. lier le css (attention pas le scss)
    c. Rédiger le HTML
4. Créer les classes BEM au fur et à mesure de la rédaction HTML
5. Tracer l'arbre BEM indépendant .md
    a. Aide pour écrire CSS
    b. Aide pour Javascript
6. Attaquer le CSS Global

    a. Mobile sans media query
7. On redimensionne le navigateur en observant la largeur en pixel
    a. activer le mode responsive de l'inspecteur
    b. on choisit la largeur de viewport qui débutera le prochain point de rupture
8. Ecrire la media query
9. On fait les modifications
    a. Apport  rendre flex un élément qui suivait le flux
    b. Modification qui consiste à écraser un CSS précédent
10. On reprend au point #7

# FLEXBOX

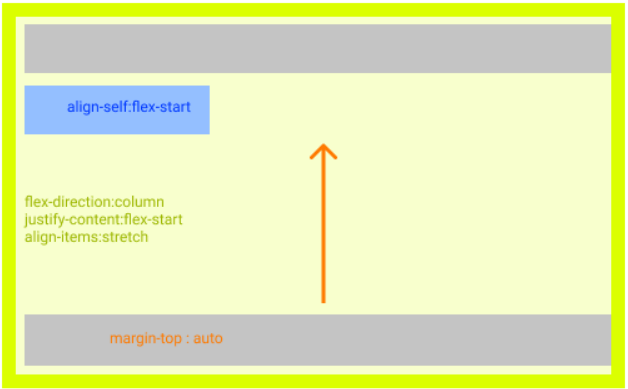
## Principe

Créer un contexte de placement flexible

- Alignement possible sur 2 axes
- Le contexte flex s'applique aux parents
    - il impacte uniquement les enfants de premier niveau
- Du point de vue extérieur le flexcontainer fonctionne comme un block
    - il existe la propriété (display:inline-flex) fonctionne comme inline-block

## Comportement des flexitems

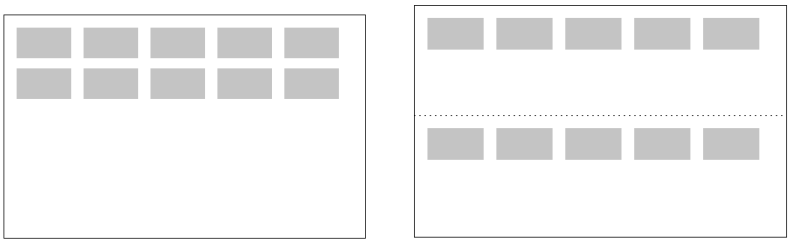
- le modèle de boîte est respecté usage des margin possible
- par défaut: se réduit à la taille du contenu sur l'axe principal
    - en revanche sur l'axe secondaire il occupe tout l'espace disponible
- Ils peuvent devenir élastiques (plus grand) grâce à flex-grow (sur l'axe principal)
- Ils peuvent se réordonner
- par défaut les éléments se placent côte à côte et débordent du container
    - forcer le retour à la ligne avec flex-wrap:wrap
- Alignements
    - Inverser l'ordre
    - répartition automatique des éléments dans le flex container (space-around / space-evenly)



## Flex container proprietés de base

- display:flex;
- flex-direction (donne l'axe principal)
- flex-wrap (retour à la ligne)
- Alignements dans le flex container
    - justify-content (axe principal)
    - align-items (axe secondaire)
        - divise le container en lignes
        - aligne l'élément dans chaque ligne
    - align-content (seulement en contexte wrap)
        - aligne dans le container

### Align items vs  align-content



# GRID SYSTEM

Créer un contexte de placement calqué sur une grille, penser à utiliser l'inspecteur de firefox pour visualiser la grille

## Fonctionnement

- s'applique au parent et impacte les enfants de premier niveau
- Attention: modèle de boîte fonctionne mais risque d'entrer en conflit avec les dimensions du motif de grille (grid-template) ainsi que les dimensions des gouttières (gap)
- Unités nouvelles
    - fr (fraction de l'espace disponible du container)
    - minmax ( valMin , valMax )

### Créer le motif de grille (sur le parent)

```
        gap:2em; //gouttières

        grid-template-columns: 100px 1fr minmax(300px , 1fr);

        grid-template-rows: 200px 300px;
```

## Particularités

### emplacement sur plusieurs cellules (proprietés sur les enfants)

- grid-columns (nombre de cellules horizontales pour le grid-item)
- grid-rows (nombre de cellules verticales occupées par le grid-item)
    - span 2 (nb de cellules)
    - numLigneDebut / numLigneFin

grid-area