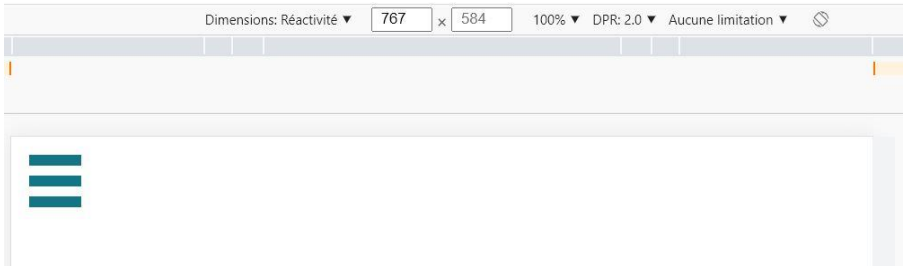
Semaine 05 - Menu "*responsive*" divers exemples et +

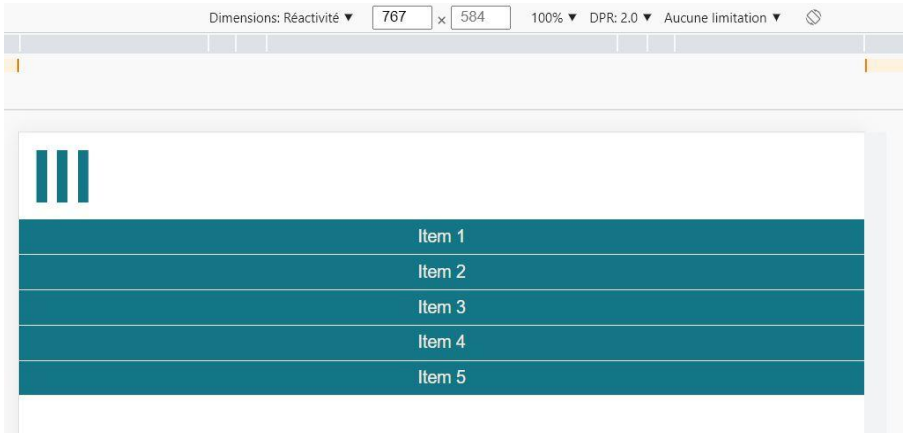
Dans le même esprit que les exemples vus à la semaine 04, nous pouvons réaliser un menu adaptatif ("*responsive*") à partir d'un élément "**input**" de type "**checkbox**", de la légende "**label**" associée et de la pseudo-classe "**:checked**" détectée sur l'élément de type "*checkbox*".

Menu "responsive" en pur HTML-CSS avec input checkbox et :checked

Dans l'exemple simple présenté ici, nous allons illustrer un menu typique d'un site web classique, afin de réduire le code CSS requis. Sur petit écran (largeur maximale de 767px), le menu sera à priori caché et on verra uniquement le bouton *burger*.



Un clic sur ce bouton fera apparaître ou disparaître le menu.



Sur plus grand écran (largeur minimale de 768px et +), on verra simplement le menu conventionnel en haut de page.

La structure HTML de notre exemple est ici très simple. Le menu est intégré dans une balise <header> avec la case à cocher et la légende (éléments **input** de type *checkbox* et **label**):

```
<body>
    <header>
        <input type="checkbox" id="cc-pour-bouton-burger" />
        <label for="cc-pour-bouton-burger" class="bouton-burger"></label>

        <nav class="menu">
            <a href="#">Item 1</a>
            <a href="#">Item 2</a>
            <a href="#">Item 3</a>
            <a href="#">Item 4</a>
            <a href="#">Item 5</a>
        </nav>
    </header>
</body>
```

Code CSS pour les petits écrans

Apparence du menu

Le principe général sur petit écran est le suivant:

- le menu est un **conteneur flex-box** affiché en **colonne**;
- On le retire du flux normal du document avec un **position fixe** *(position:fixed;)*. ***Il est à noter ici que la stratégie de retirer le menu du flux normal de la page HTML avec des positons fixes ou absolues est très utilisée par les concepteurs Web pour les petits écrans***;
- Sur l'axe des Y on lui donne une position avec la **propriété top**;
- Pour donner l'impression que le menu est masqué, on lui donne une **hauteur maximale de 0%**. ***À noter que différentes stratégies peuvent être utilisées pour retirer le menu de l'affichage, c'est ce que nous verrons dans les différents exemples de la semaine 05***;
- Et, on cache le contenu qui déborde, notamment les éléments <a> du menu avec la gestion de la propriété overflow. ***Pas toujours nécessaire, selon la stratégie utilisée***.

Le code CSS du menu est le suivant:

```
.menu {
    width: 100%;
    display: flex;
    flex-direction: column;
    /*Position fixe*/
    position: fixed;
    left: 0;
    /*Position sous le bouton burger avec une hauteur maximale de 0%*/
    top: 5rem;
    max-height: 0%;
    /*On masque le contenu qui déborde, soit les boutons*/
    overflow: hidden;
    /*Effet de transition sur la hauteur maximale*/
    transition: max-height 1s;
}
```

Contrôle de l'affichage du menu avec le bouton burger

Le principe général pour le contrôle du menu avec le bouton burger est le suivant: avec la *pseudo-classe* **:chekded** on vérifie si on a cliqué sur la case à cocher ayant ici le **id: #cc-pour-bouton-burger**, et ce, même si celle-ci est retirée de l'affichage. Si oui, on contrôle les **deux éléments HTML voisins** suivant:

- on fait tourner le bouton burger ayant la classe: **.bouton-burger**;
- on donne une hauteur maximale au menu ayant la classe : **.menu**.

Le code CSS est le suivant:à

```
/*Gestion du bouton burger si la case à cocher est cochée*/
#cc-pour-bouton-burger:checked~.bouton-burger {
    transform: rotate(90deg);
}

/*Apparition du menu si la case à cocher est cochée*/
#cc-pour-bouton-burger:checked~.menu {
    max-height: 100%;
}

/* Masquage de la case à cocher*/
#cc-pour-bouton-burger {
    display: none;
}
```

Code CSS pour les écrans plus grands

Enfin, il reste à gérer l'aspect du menu pour les plus grands que 768px pour cet exemple. Pour cela:

- on enlève la position en absolue du menu, pour le remettre à la valeur de positionnement par défaut (*static*);
- en tant que conteneur flex, on met l'axe principal d'affichage du menu en rangée (*row*);
- les éléments du menu peuvent s'étirer pour prendre toute la place disponible;
- et finalement, on enlève de l'affichage le bouton burger.

Le code CSS est le suivant:

```
@media screen and (min-width: 768px) {
    .menu {
        position: static;
        flex-direction: row;
    }

    .menu a {
        flex: 1 1 0%;
    }

    .bouton-burger {
        display: none;
    }
}
```

Quelques ajustements nécessaires avec JavaScript

En théorie, tout est fonctionnel avec le code CSS du menu adaptatif ci-haut, sauf qu'on peut être confronté aux deux problèmes suivants sur les petits écrans:

- Le premier, est qu'on pourrait continuer à faire défiler la page Web, même si le menu est affiché en avant-plan;
- Le second, est qu'en cliquant sur un élément du menu qui devrait nous diriger à un endroit différent de la page Web, le menu ne se ferme pas.

Le CSS, ne suffit malheureusement pas à solutionner ces deux problèmes. Il faut introduire ici un peu de JavaScript.

Contrôler le défilement de la page avec JavaScript si le menu est affiché sur petit petit écran

Sur petit écran, il peut être requis d'empêcher le défilement de la page Web lorsque le menu adaptatif est affiché. Pour cela, on doit réaliser les étapes suivantes:

- Récupérer la case à cocher avec la méthode document.querySelector("*selecteurCSSValide*") que nous connaissons déjà;
- Récupérer l'élément <body> de la page aussi avec la méthode document.querySelector("*selecteurCSSValide*");
- Modifier par programmation JavaScript la propriété "overflow" de l'élément <body> pour mettre sa valeur à "hidden" lorsqu'un clic sur la case à cocher est détecté. **À noter qu'on pourrait aussi modifier une autre propriété selon les besoins du projet**, comme de mettre une position fixe (position:fixed) à l'élément <body>.

Le code JavaScript requis, pourrait ressembler à ceci:

```
//Récupérer la balise <body>
let leCheckBox = document.querySelector("#cc-pour-bouton-burger");
let leBody = document.querySelector("body");

leCheckBox.addEventListener("click", gererLeDefilement);

function gererLeDefilement(event) {
    if (leCheckBox.checked == true) {
        leBody.style.overflow = "hidden";
    } else {
        leBody.style.overflow = "scroll";
    }
}
```

Gérer l'état de la case à cocher pour fermer le menu

Lorsqu'on clique sur un bouton du menu, on doit s'assurer de pouvoir fermer le menu, mais aussi de remettre le défilement possible de la page si on l'a enlevé à l'étape précédente (*ou de réinitialiser tout autre propriété modifiée*). Pour cela on doit réaliser les étapes suivantes:

- Récupérer tous les boutons du menu avec une nouvelle méthode qui est: document.**querySelectorAll**("selecteurCSSValide"). Cette méthode renvoie **une liste des éléments** du document qui correspondent au groupe de sélecteurs spécifiés;
- Et, **mettre un gestionnaire d'événement sur chaque bouton** avec une **boucle *for…of*** afin de gérer la valeur de la propriété "*checked*" de la case à cocher. Soit directement, ou indirectement en appelant la méthode **click()** de la case à cocher.

Le code JavaScript requis, pourrait ressembler à ceci:

```
/*Récupérer la case à cocher et la balise <body>*/
let leCheckBox = document.querySelector("#cc-pour-bouton-burger");
let leBody = document.querySelector("body");

/*Récupérer les boutons du menu*/
let lesBoutons = document.querySelectorAll(".menu a");

/*Mettre un gestionnaire d'événement sur chaque bouton avec un boucle for of*/
for (let unBouton of lesBoutons) {
    unBouton.addEventListener("click", controlerBoutonBurger);
}

function controlerBoutonBurger() {
    //Gérer l'état de la case à cocher
    leCheckBox.checked = false;

    //Remettre la barre de défilement sur le body
    leBody.style.overflow = "scroll";

    //Ou :
    /*
      // Cliquer programmatiquement la case à cocher, mais seulement si déjà cochée
      if(leCheckBox.checked == true) {
         leCheckBox.click();
      }
    */
}
```

Quelques références utiles sur les sujets de la semaine 05

- Chartier, Mathieu (mai 2021). Astuces en HTML-CSS avec input checkbox et :checked. repéré à: https://blog.internet-formation.fr/2021/05/astuces-en-html-css-avec-input-checkbox-et-checked/
- Mozilla Corporation (Novembre 2022). Document.querySelectorAll() - Référence Web API | MDN. repéré à: https://developer.mozilla.org/fr/docs/Web/API/Document/querySelectorAll