

## T.P. 4 - **Kampadejo** - une navigation accessible

### Mandat

- Compléter la schématisation de la mise en page.
- Intégrer et adapter à la page web fournie **un lien rapide** vers le contenu, **une navigation par régions** ainsi qu'**un menu responsive accessible**.
- Versionner **localement** seulement. Débuter le versionnage avec la commande `git init`.

### Étapes

#### 1. Schématisation

**Cette tâche vous permettra d'analyser les stratégies d'intégration mises en œuvre dans ce projet.**

Afficher dans un navigateur, le fichier index.html du dossier sources.

Ouvrir l'inspecteur du DOM.

En parallèle, ouvrir le schéma fourni (schématisation-a-compléter.pdf) dans Photoshop pour pouvoir ajouter du texte... (Alternativement, vous pouvez imprimer le document, le compléter au crayon puis le numériser).

Compléter le document en identifiant dans le haut de chaque encadré **les éléments html** qui agissent comme conteneurs de centrage, de rangées ou de blocs.

Ajouter au nom de l'élément html, ses **classes** du système de grilles fluides et de contexte.

Exemple pour le premier encadré qui regroupe les rangées 1 et 2 : **header.entete**.

**Faire une remise formative de cette étape avant de poursuivre.**

#### 2. Intégrer le html manquant pour les mécanismes de navigation

##### 2.1 Ajouter le meta viewport

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

##### 2.2 Lien rapide aussi appelé lien d'évitement

Tout de suite après l'ouverture de la balise **<body>**, le lien rapide "Allez au contenu"

```
<a href="#contenu" class="sauter screen-reader-only focusable">Allez au contenu</a>
```

La cible de ce lien est défini par l'attribut id du premier h2 : *Notre entreprise*.

Vérifier le fonctionnement du lien rapide : il doit apparaître lorsqu'il reçoit le focus en entrant dans la page, à la première touche tab et la touche *Enter* doit permettre de naviguer au contenu.

##### 2.3 Menu des régions

Il y a seulement trois régions à définir dans ce document :

- le bandeau d'entête : balise **<header>** et role **banner**,
- le contenu principal : balise **<main>** et role **main**,
- le pied de page : balise **<footer>** et role **contentinfo**.

## 2.4 Menu principal

Compléter le html pour que votre menu soit balisé de manière similaire à celui du cadriciel de votre guide de développement.



Dans la balise `header`, repérez où ajouter la balise `nav` du menu principal.

La balise `nav` doit avoir les classes `"nav"` et `"nav--closed"`.

La balise `nav` doit avoir comme enfants :

- Une balise `a` avec la classe `"logo"`. Cette balise `a` contient une balise `img`.
- Un `ul` avec la classe `"nav__list"`. Ce `ul` doit contenir 4 items de liste (`li`).

Avec Emmet

Pour générer ce HTML, ça donne en format abrégé : `nav>a+ul`

De manière plus complète (à mettre bout à bout en enlevant les espaces et les sauts de ligne) :

```
nav.nav.nav--closed>(a.logo[href="index.html"]>img[src="images/logo.svg"][alt="Kampadejo"])+(ul.nav__list>li.nav__list-item*4>a.nav__link)
```

Ajuster les contenus html et les valeurs des attributs href des liens du menu principal.

Les liens doivent pointer vers les différentes sections de la page :

- Entreprise
- Expériences
- Services
- Contact

## 3. Compléter l'intégration du menu principal

### 3.1 Définir les styles de base du menu et du logo

Dans le fichier `styles.css`, ajouter des styles à la section

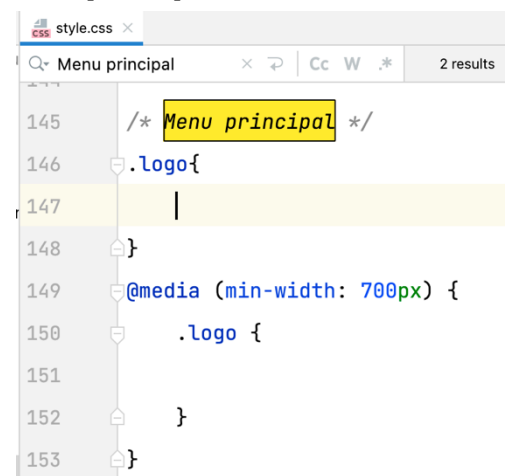
```
/* Menu principal */
```

#### Dimensionner le SVG du logo

- Ajouter un `display:inline-block` au sélecteur `.logo` et fixez le `width` à 150px sur l'écran étroit et 180 sur l'écran large

#### Positionner le logo à gauche et le menu (ou son bouton) à droite

- Ajouter au sélecteur `.logo` la règle `float:left`
- Ajouter un sélecteur `.nav` avec la règle `text-align:right;`



### 3.2 Lier le JavaScript

Dans le html, ajouter la balise script à la fin du document, juste avant la fermeture du **<body>** :

```
<script src="js/menu.js"></script>
</body>
```

### 3.3 Tester le menu

Le bouton menu n'apparaît que sur l'écran étroit et le menu est fermé au départ.

Le menu doit être accessible même si JavaScript est désactivé.

Le menu doit être facilement navigable au clavier : ouverture et fermeture du menu mobile, effets de focus, etc. Il faudra refaire ces derniers tests une fois les styles d'interactivité complétés !


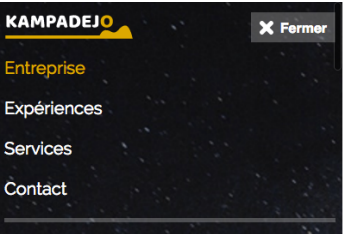

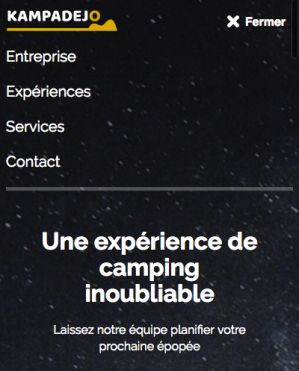
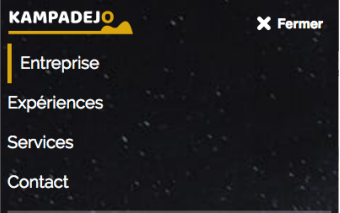
### 3.4 Ajouter les styles d'interactivité du menu

Les fichiers menu.css et menu.js proviennent du cadriceil.

Vous devez redéfinir les styles du **menu.css**, selon la direction visuelle de Kampadejo.

Les icônes de hamburger et de X seront ajoutés à l'étape 4.

#### Sur l'écran étroit

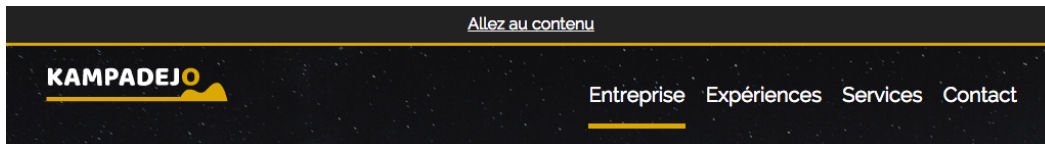
Fermé / Ouvert	Survol	Focus
	<p><b>Bouton de menu</b></p> <ul style="list-style-type: none"> <li>- fond gris transparent   <code>rgba(255, 255, 255, 0.25)</code></li> </ul> <p><b>Items de menu</b></p> <ul style="list-style-type: none"> <li>- couleur jaune   <code>#dba70d</code></li> </ul> 	<p><b>Bouton de menu</b></p> <ul style="list-style-type: none"> <li>- Outline jaune</li> </ul> 
		<p><b>Items de menu</b></p> <ul style="list-style-type: none"> <li>- bordure latérale jaune</li> </ul> 

### Sur l'écran large

Au survol : couleur jaune



Au focus : bordure inférieure jaune sur les items de menu



## FIN DE LA PREMIÈRE ÉTAPE – pour mercredi le 10 avril

### 4. Ajouter les visuels au format SVG

#### (1) Ajouter les icônes du bouton Menu / Fermer

À l'aide d'un pseudo-élément `:before`, ajouter en images d'arrière-plan les icônes au bouton de menu.

Les fichiers sont *ico-fermer.svg* et *ico-menu.svg*.

Le changement d'icône sera provoqué par l'ajout ou suppression de la classe `nav--closed` par le JavaScript sur la balise `nav`. Repérer les sélecteurs à compléter dans le fichier `_menu.css`.

```
.nav__control:before{
  display: inline-block;
  content: "";
  width: 1em;
  height: 1em;
  /* compléter pour afficher l'icône X */
}

.nav--closed .nav__control:before{
  /* compléter pour afficher l'icône Hamburger */
}
```

Remarquez les propriétés (`display`, `content`, `width` et `height`) utilisées pour former la boîte du pseudo-élément `:before`.

#### Astuce :

Pour que l'espace de l'image d'arrière-plan reste proportionnée avec la taille de police du libellé, au lieu d'utiliser des pixels on utilise la valeur `1em`. Ainsi l'icône aura toujours la même grandeur que la taille de police puisque les `em` multiplie le `font-size` hérité du parent.

#### (2) Ajouter les icônes des médias sociaux dans le pied de page

Commencer par cacher visuellement les libellés courriel, téléphone et facebook en ajoutant à leur `<span>` la classe `screen-reader-only`.

Les classes (`icone`, `icone-courriel`, `icone-telephone`...) permettant d'afficher les icones sont déjà définies au début du fichier `style.css`. Ajouter le HTML nécessaire pour afficher chaque icône.

## 5. Déclarer quelques variables dans le fichier utilitaires.css

Repérer les valeurs qui se répètent dans le fichier et faites-en des variables.

Par exemple, la couleur du texte de base, la couleur du texte au survol, la famille de police et les valeurs des bordures apparaissant au focus :

```
:root {  
  --color-text: #fff;  
  --color-Y: #dba70d;  
  --color-G: rgba(255, 255, 255, 0.25);  
  --font-family: Raleway, sans-serif;  
  --border: 3px solid #dba70d;  
}
```

Pour utiliser ces variables, il suffit de la placer dans une instruction var(). Exemple :

```
.btn:hover {  
  background-color: var(--color-Y);  
  border-color: var(--color-Y);  
  color: #000;  
}
```

### Modalités de remise

Remise sur Léa, un dossier **tp4-*initiales*PrenomsNOMsEtudiant.zip**  
contenant le dossier Web SANS le dossier \_enonce-guides

### Échéance

Groupes 1 et 2 : Jeudi le 11 avril 23h59

## Grille d'évaluation

Ce travail pratique vaut pour 5-10% de la session

<b>Schématiser l'intégration avec un système de grilles fluides</b>	<b>1</b>
Définir correctement les conteneurs html pour le centrage, les rangées et les blocs. Définir correctement les classes à utiliser pour réaliser la mise en page en respect de la grille de 12 colonnes.	
<b>Intégrer les contenus multimédias en respect des meilleures pratiques d'accessibilité, de performance et de portabilité.</b>	<b>1</b>
Intégrer un lien rapide « aller au contenu ». Intégrer une navigation par régions (banner / main / contentinfo / complementary .	
<b>Intégrer un patron de conception de menu responsive, accessible et en respect des principes de l'amélioration progressive.</b>	<b>2</b>
Par défaut, le menu est déployé et il n'y a pas de mécanisme propre à la version enrichie en javascript. Les changements d'apparence se font par des bascules de classes. Les styles propres à la version enrichie sont conditionnels à la classe "js". Les étiquettes des mécanismes (<button>) sont claires et lisibles. Par exemple : le texte « fermer » est présent dans le button, l'information ne repose pas seulement sur la présence d'un icône « X ». L'interactivité du menu est fonctionnelle et conforme au démo fourni (le cadriciel). Les liens sont fonctionnels.	
<b>Styler un menu responsive. Styler l'interactivité.</b>	<b>3</b>
Le positionnement du menu a été adapté en respect des captures écrans fournies pour correspondre aux variantes pour écran étroit et pour écran large. Les styles des différents états du menu et des hyperliens ont été complétés en respect des captures-écrans fournies pour les états ouvert et fermé, survol et focus.	
<b>Compléter l'intégration. Optimiser le code par l'ajout de variables.</b>	<b>2</b>
Dimensionner le logo du menu. Ajouter les icônes hamburger et x au bouton de menu. Ajouter les icônes de medias sociaux. Ajouter et utiliser des variables.	
<b>Versionner le travail pratique localement avec GIT</b>	<b>1</b>
Minimum de quelques commit et qualité des étiquettes de commit.	
<b>Total</b>	<b>10</b>