Rappel : pour toute la durée du module, le travail attendu est un TRAVAIL PERSONNEL.

Partie 1 – Étude d'une page fournie

Analyser la page tp3part1.html fournie sur Chamilo (dans le zip TP3).

Expliquer (en détail !) :

- A quoi sert la balise <style> incluse dans l'entête, et comment sont structurées les directives qu'elle contient (à quoi elles s'appliquent, ce qu'elles font…)
- Pourquoi le titre de la page est rose alors que la première directive définit que le texte doit être bleu.
- Quelles sont les différences de rendu entre « Mon premier chapitre » et « Mon second chapitre » (apparence, taille,…)
- Quelles sont les deux approches qui ont permis d'arriver à des rendus si similaires ? Quelles sont les limites de chacune des deux approches ? Laquelle préféreriez-vous et pourquoi ?
- Dans la première section du premier chapitre, le texte est présenté sur trois colonnes. Les directives CSS appliquée à .detail définit cette présentation, mais… à quoi servent respectivement les directives column-count, column-gap et column-rule ? Pourquoi sont-elles indiquées chacune 3 fois, avec –moz- ou –webkit- devant ?
- Pourquoi la première lettre de chaque paragraphe dans ces colonnes est-elle présentée différemment ? Comment le modifieriez-vous pour que la lettre soit un peu décalée sur la droite ?
- Retirez les directives du fichier HTML et copiez les dans un fichier indépendant. Ajouter les balises nécessaires pour que ce fichier CSS soit utilisé par votre fichier HTML. Quels sont les avantages de cette approche ?

PARTIE 2 – Ajout de style sur une page existante

En modifiant le code TP3.html (renommé en TP3.txt) fourni sur Chamilo, ajouter directement dans le code de la page (en section head) les directives CSS nécessaires pour que :

- le texte de toutes les cellules de lignes de classe bestOf dans la table d'identifiant lesMeilleursVendeurs soit affiché en vert (sans changer celles de la table d'identifiant lesMeilleursProduits).
- le texte de toutes les cellules d'une ligne de classe worstOf, quel que soit son parent, s'affiche en rouge.
- le texte de la première ligne ayant la classe bestOf s'affiche en gras.
- sans qu'elle devienne prioritaire sur les directives précédentes, ajouter UNE directive pour que les lignes paires des tableaux aient un texte bleu.
- les développeurs de la page se rendent compte qu'ils ont oublié d'indiquer l'unité monétaire du chiffre d'affaire : créer une directive CSS pour rajouter "€" derrière les chiffre d'affaire des vendeurs et des produits (mais pas ceux des services).
- dans le formulaire de contact, rajouter de l'ergonomie en…
    - regroupant les champs : ajouter un fieldset contenant mail et téléphone, avec comme légende "Vous répondre", et un second fieldset contenant objet et message, avec comme légende "Votre message" ;
    - alignant les libellés à **droite** et en haut (textarea)

    - sur une petite surface d'affichage, les libellés doivent s'afficher au dessus des champs et non à côté
    - centrant le bouton "Envoyer"
    - rendant obligatoire les champ "mail" et "message" ; mettez leur couleur de fond en rouge pâle lorsqu'ils sont vides pour les rendre évidents (à faire en UNE directive CSS avec UN sélecteur, qui doit marcher pour tous les champs obligatoires, présents et à venir)
- trouver trois directives différentes (trois types de sélecteurs) pour que le titre de la section d'identifiant "services" soit affiché en jaune
- réduisez la page à la taille d'un écran de smartphone : elle ne ressemble à rien… Modifier tout le code nécessaire pour que :
    - sur un PC (largeur d'écran > 900px) les 3 premières sections se trouvent côte à côte avec le formulaire centré en dessous
    - sur une tablette (900px > largeur d'écran > 480px), les deux premières sections se trouvent côte à côte, tandis que la troisième et le formulaire sont en dessous, eux aussi côte à côte, sur toute la largeur de l'écran
    - sur un smartphone (480px >= largeur d'écran), les quatre sections se trouvent les unes en dessous des autres et occupent toute la largeur d'écran.

Partie 3 – Application à votre site

Nous allons maintenant revenir sur votre site de présentation de recettes et/ou de Pokemon, pour y ajouter du style CSS.

ATTENTION : *tous les styles doivent être écrits en CSS ! (balises <b>, <i>, … interdites). Les éléments de même nature doivent pouvoir être modifiées en un seul changement dans les directives CSS. Par exemple, tous les ingrédients doivent dépendre de la / des mêmes directives CSS. De même, les descriptions des recettes / Pokemon doivent pouvoir être modifiées toues d'un seul changement dans le CSS. A l'inverse, des rubriques différentes (ingrédients et préparation) peuvent avoir le même rendu mais doivent pouvoir être stylisées différemment en ne modifiant QUE le CSS (d'ailleurs, si vous n'êtes pas coutumiers de cette structuration, ce serait bon de réfléchir à son intérêt et à ses limites…).*

Etapes :

1. Ajouter des directives CSS à vos pages. Essayez de les rendre à la fois attractives et avec un code efficace (compact, mais maintenable, c'est-à-dire factorisé quand c'est intelligent et commenté…). Vous êtes libre du style, mais vous devez avoir au minimum :
    - Des styles pour les titres
    - Des styles différents pour des rubriques différentes
    - Une gestion de positionnement
2. Prévoir :
    i) Une feuille de style pour l'affichage à l'écran d'un PC
    ii) Une feuille de style pour l'affichage sur un Smartphone
    iii) Une feuille de style pour l'impression

    Nota : vous pouvez faire de l'esthétique si vous voulez, mais l'objectif ici est d'arriver à ce que les différences sautent aux yeux entre les trois versions et elles doivent concerner aussi bien l'esthétique (les couleurs, polices,…) que l'organisation de la page (exemple : présentation en ligne sur PC, en colonne sur téléphone, image à droite de la page sur PC, au centre pour l'impression etc.)

3. Menu statique

Enfin, sur chaque page, vous allez intégrer un « menu ». Ce menu sera divisé en 2 colonnes. La colonne de gauche offrira une seule option (cliquable) : Accueil (retour à la page d'accueil du site).

La seconde colonne comportera un texte (ex : Recettes, Pokemons,…) et des liens vers les 3 pages de présentation (avec par exemples 3 noms de recette ou de Pokemon), quelque chose comme :

Accueil	**Entrées**	**Plats**
	Salade du chef	Poulet Basquaise
	Œufs Mimosa	Veau Marengo
	Avocats aux crevettes	…

Ce menu doit

- Etre placé en haut de chaque page, occupant toute la largeur, et avoir un fond d'une couleur qui masque ce qui est derrière lui
- Il doit rester à cette place même si l'on fait défiler la page (le menu doit rester visible)
- Avoir l'option active (le lien Accueil lorsque la page d'accueil est affichée !) visuellement identifiable.
  Etudiez ensuite comment « automatiser » ce rendu visuel en ayant pour le menu **le même** code HTML et les mêmes directives CSS dans toutes les pages (l'idée : peut-on externaliser le menu et son style pour les factoriser ?)

Vous devez vous inspirer de la structure HTML suivante :

```
<div class="menuItem" id="identifiantDuMenu1">
        Entête du menu 1
        <ul>
                <li class="menuElement"><a href="...">1er élément du menu1</a></li>
                <li class="menuElement"><a href="..">2ième élément du menu1</a></li>
                        ...
        </ul>
</div>
<div class="menuItem" id="identifiantDuMenu2">
        Entête du menu 2
        <ul>
        <li class="menuElement"><a href="...">1er élément du menu2</a></li>
        <li class="menuElement"><a href="...">2ième élément du menu2</a></li>
                        ...
        </ul>
</div>
```

4. Menu déroulant : transformer le menu statique du haut du CV en menu déroulant en utilisant uniquement du CSS. Exemple :

Accueil	**Entrées**	**Plats**
	Salade du chef	
	Œufs Mimosa	
	Avocats aux crevettes	

Évidemment, le code CSS ne doit pas dépendre du nombre d'options ou de sous-options du menu et ne doit pas être spécifique pour une option donnée (en clair : pas de directive dépendant d'un id).