Galerie d'exercices

HTML CSS JS

Bouton change color & contenu

- Créer une page interactive où un bouton permet de changer dynamiquement la couleur et le contenu d'une section.
- ### HTML :
- Créer une section principale `.full-view` qui occupe toute la hauteur de la fenêtre (100vh).
- Ajouter un titre `<h2>` avec l'identifiant `section-heading`.
- Ajouter un paragraphe `<p>` avec l'identifiant `section-paragraph`.
- Ajouter un bouton `<button>` avec l'identifiant `changeContentButton`.
- ### CSS :
- Appliquer une couleur de fond bleue et une couleur de texte blanche à `.full-view`.
- Centrer verticalement et horizontalement le contenu avec Flexbox.
- Styliser le bouton avec :
- - coins arrondis,
- - couleurs complémentaires,
- - effet de transition au survol.
- ### JavaScript :
- Sélectionner tous les éléments nécessaires (`section`, `h2`, `p`, `button`).
- Créer une variable `isOriginal` pour suivre l'état du contenu.
- Ajouter un événement `click` sur le bouton qui :
- - change le fond en vert et le texte s'il est en état original,
- - ou restaure les valeurs bleues d'origine sinon,
- - et inverse l'état à chaque clic (`isOriginal = !isOriginal`).

Sélecteurs et déclarations

- 1. Créer un fichier CSS et le lier au fichier HTML.
- 2. Créer un élément h1 et une liste ordonnée.
- 3. Assigner au sélecteur h1 une couleur verte et une taille de 50px.
- 4. Assigner au sélecteur ol une couleur rouge et le mettre en gras.

Sélecteurs et déclarations avec ID et classe

- 1. Créer un paragraphe.
- 2. Créer un deuxième paragraphe, lui donner une `class`, lui assigner une bordure de 4px d'épaisseur, en pointillé et bleu clair et assigner à ce même paragraphe, une opacité de 0.5.
- 3. Assigner les valeurs de cette nouvelle CLASS au premier paragraphe.
- 4. Le premier paragraphe est centré.

Sélecteurs et déclarations CSS

- 1. Créer un fichier CSS, le lier au fichier HTML.
- 2. Créer un élément h1 et une liste ordonnée.
- 3. Assigner au sélecteur h1 une couleur verte et une taille de 50px.
- 4. Assigner au sélecteur ol une couleur rouge et le mettre en gras.

ID et class en CSS

- 1. Créer un paragraphe.
- 2. Créer un deuxième paragraphe, lui donner une `class`, lui assigner une bordure de 4px d'épaisseur, en pointillé et bleu clair et assigner à ce même paragraphe, une opacité de 0.5.
- 3. Assigner les valeurs de cette nouvelle `class` au premier paragraphe.
- 4. Le premier paragraphe est centré.

L'héritage et la priorité

- 1. Créer une page HTML avec un style CSS.
- 2. Ajouter des styles de couleur et vérifier les comportements d'héritage et de priorité avec les éléments enfants.

Les types d'éléments en CSS

- 1. Créer une page HTML avec des éléments de type `block`, `inline` et `inline-block`.
- 2. Utiliser la propriété `display` pour changer leur type et vérifier le comportement.

🛒 Boutique en ligne (FakeStoreAPI)

- 🎯 Objectif : fetch, then, DOM, createElement, insertion dynamique.
- 🟢 Niveau 1 – Affichage des données dans la console
- Fais une requête fetch() vers l'API et affiche les données dans la console.
- Affiche dans la console tous les titres des produits.
- ```js
- fetch("https://fakestoreapi.com/products")
- .then(response => response.json())
- .then(data => {
- // affiche ici tous les titres des produits
- });
- ```
- 🟡 Niveau 2 – Création dynamique des cartes
- Crée une div avec l'id="product-container" dans ton HTML.
- Affiche chaque produit sous forme de carte contenant :
- l'image (product.image)
- le nom (product.title)
- le prix (product.price)
- 🟥 Niveau 3 – Style CSS de base
- Ajoute du style dans style.css :
- des cartes blanches avec ombre légère,
- image en haut, titre et prix en dessous,
- conteneur central en flex-wrap avec gap entre les cartes.
- Bonus : effet hover (agrandissement léger au survol).
- 🟥🟥 Niveau 4 – Filtrage par catégorie
- Affiche uniquement les produits de la catégorie "jewelery" (ou une autre de ton choix).
- Deux méthodes possibles :
- Utilise directement l'URL : https://fakestoreapi.com/products/category/jewelery
- ou filtre en JS avec `.filter(product => product.category === "jewelery")`
- ✅ Objectif final
- Tu dois obtenir une mini boutique avec plusieurs cartes dynamiques stylées, et une requête API propre.