Galerie d'exercices

HTML CSS JS

Variables et Constante (1/2)

- Créer un fichier Index.html, un fichier script.js et relier les deux en ajoutant une balise script dans le body de l'Index.html.
- Dans le fichier script.js, on voudra stocker plusieurs valeurs:
- Le nom
- Le prenom
- L'âge
- Le numero de Sécurité sociale
- Indiquer si chacune de ces valeurs sera stockée dans une variable ou une constante et les déclarer dans le fichier.

Variables et Constante (2/2)

- Lancer l'Index.html en live server.
- Ouvrir la console du Navigateur.
- Afficher la valeur de la variable age.
- Changer la valeur de la variable age.
- Vérifier que la modification s'est effectuée

Joyeux anniversaire - Chaînes de caractères - Concaténation

- 1. Créer trois variables (nom, prenom et age).
- 2. Utiliser la console pour faire apparaitre le nom le prénom et l'âge stockés (avec des espaces entre chaque) en concaténant.
- 3. Utiliser la console pour Ecrire : "Joyeux Anniversaire" suivit du nom, du prénom et de "Vous avez" age "aujourd'hui!".
- BONUS:
- Créez une section ou un élément HTML (par exemple un <p> avec un id) dans le fichier index.html, où le texte sera inséré.
- Dans le fichier script.js :
- - Sélectionnez cet élément en utilisant son id via JavaScript.
- - Insérez dans cet élément une phrase sous la forme : "J'ai [âge] ans." en utilisant la variable age.
- Vérifiez que la phrase s'affiche correctement dans la page lorsque vous l'ouvrez en Live Server.

Moyenne - Les nombres

- 1. Déclarez trois variables note1, note2 et note3 et assignez-leur des nombres représentant des notes. Ensuite, calculez la moyenne de ces trois notes et stockez le résultat dans une variable moyenne. Affichez la moyenne dans la console.
- 2. Déclarez une variable nombreTotal représentant une quantité totale, par exemple 100. Ensuite, déclarez une variable pourcentage représentant un pourcentage de cette quantité totale, par exemple 20 pour 20%. Calculez le pourcentage de la quantité totale et stockez le résultat dans une variable resultat. Affichez le résultat dans la console.
- BONUS
- Dans le fichier HTML :
- Ajoutez une balise <p> avec un identifiant (par exemple id="p1") qui servira de conteneur pour afficher la moyenne calculée.
- Dans le fichier JavaScript :
- Utilisez la méthode document.getElementById('p1') pour sélectionner la balise <p> que vous avez créée.
- Ensuite, modifiez son contenu en utilisant la propriété .innerHTML pour y insérer la phrase : "La moyenne des notes est : [valeur calculée]".

Réussite exam - Opérateurs de comparaison et condition if

- 1. Déclarez une fonction appelée verifierReussiteExamen qui prend un paramètre note.
- 2. À l'intérieur de la fonction, utilisez une instruction conditionnelle pour vérifier si la note de l'étudiant est supérieure ou égale à la note minimale pour réussir l'examen.
- 3. Définissez la note minimale pour réussir l'examen à 10.
- 4. Si la note est supérieure ou égale à 10, affichez un message indiquant que l'étudiant a réussi l'examen.
- 5. Sinon, affichez un message indiquant que l'étudiant a échoué à l'examen.

Fonctionnalité de salutations - Fonctions

- 1. Créer une variable username et mettez-y la valeur de votre choix (de type string).
- 2. Créer une variable jours et mettez-y la valeur de votre choix (de type number).
- 3. Créer une fonction salutations(), qui prend en paramètres username et jours et qui incrémentera jours de 1 et indiquera dans la console "Bonjour ... , vous êtes connectés depuis ... jours".
- 4. Appelez salutations() et vérifiez votre console.

Gestion stock magasin v1

- Vous gérez un magasin en ligne et vous souhaitez créer des identifiants de produit uniques en combinant un numéro de produit avec un libellé. Vous avez un numéro de produit initial et un libellé de produit.
- 1. Déclarez une variable numeroProduit et assignez-lui un nombre représentant le numéro initial du produit.
- 2. Déclarez une constante libelleProduit et assignez-lui une chaîne de caractères représentant le libellé du produit.
- 3. Utilisez une opération mathématique pour augmenter le numeroProduit de 1.
- 4. Passez libelleProduit en majuscules et stockez la valeur finale dans une variable.
- 5. Concaténez le numeroProduit avec le libelleProduit en majuscules pour former l'identifiant unique du produit.
- 6. Affichez l'identifiant unique du produit dans la console.

Jeu du Guess The Number v1

- Créer un jeu "Devine Le Nombre" :
- L'ordinateur choisit un nombre au hasard entre 1 et 10 et l'utilisateur doit deviner ce nombre.
- Le jeu choisira un nombre aléatoire entre 1 et 10 (inclus).
- 1. Tu as un seul essai pour deviner ce nombre.
- 2. Utilise la fonction prompt() pour saisir ta proposition de nombre.
- 3. Si ton nombre est égal au nombre mystère, tu gagnes ! Sinon, le jeu affichera le nombre mystère.

Nein nein - Itérations

- 1. Créez une fonction sayHello qui affichera dans la console "Bonjour" suivi du prénom de chaque apprenant de la cdweb. Utilisez une boucle pour ça.
- 2. Créez une boucle qui executera 10 fois le texte "NEIN".

Jeu du Guess The Number v2 (avec ou sans modif du DOM)

- Améliorer le jeu du "Devine Le nombre" :
- Le joueur a le droit à 3 essais. S'il dépasse ce nombre, le jeu recommence.
- 1. Le jeu choisira un nombre aléatoire entre 1 et 10 (inclus).
- 2. Tu as trois essais pour deviner ce nombre.
- 3. Utilise la fonction prompt() pour saisir ta proposition de nombre.
- 4. Si ton nombre est égal au nombre mystère, tu gagnes ! Sinon tu dois retenter ta chance. Au bout de trois essais, tu as perdu et le jeu recommence.
- Bonne chance !

Tableau à deux dimensions - Affichage des matières et notes

- Créer une interface qui utilise un tableau à deux dimensions pour afficher une liste de matières et leurs notes. Ajoutez une fonctionnalité pour calculer et afficher la moyenne des notes.
- Utilisez un tableau à deux dimensions pour stocker les matières et les notes.
- - Exemple : [["Maths", 15], ["Français", 12], ["Histoire", 14], ["Physique", 17]].
- Affichez dynamiquement les matières et les notes dans un tableau HTML.
- Ajoutez un bouton "Calculer la moyenne".
- Quand l'utilisateur clique sur ce bouton, affichez la moyenne des notes.

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`).

Memory Flip Cards

- Créer un jeu où des cartes retournables doivent être associées par paires. Lorsqu'une paire correcte est trouvée, les cartes disparaissent. Sinon, elles se retournent après une brève période.
- ## Fonctionnalités attendues :
- 1. Chaque carte affiche une couleur lorsqu'elle est retournée.
- 2. Les cartes sont disposées aléatoirement.
- 3. Si deux cartes retournées correspondent, elles sont marquées comme "trouvées" et disparaissent.
- 4. Si elles ne correspondent pas, elles se retournent après un délai.
- 5. Une seule paire peut être retournée à la fois (verrouillage du plateau).
- ### Etapes
- Etape 1
- 1. Créer une structure HTML comprenant :
- - Un conteneur pour afficher les cartes.
- - 12 cartes avec des paires de couleurs (6 paires au total).
- - Chaque carte contient un attribut `data-color` pour stocker sa couleur.
- Etape 2
- - Créer un plateau de jeu avec une disposition en grille.
- - Style par défaut des cartes :
- - Carrées, grises, et avec des bords arrondis.
- - Change de couleur lorsqu'elles sont retournées.
- - Ajouter des classes pour les états :
- - `.flipped` : indique qu'une carte est retournée.
- - `.matched` : indique qu'une paire est trouvée (et cachée).
- Etape 3
- **Sélection des éléments HTML**
- - Récupérer toutes les cartes via leur classe.
- - Initialiser les variables `firstCard`, `secondCard` et `lockBoard` pour gérer l'état du jeu.
- Etape 4
- - **Événements de clic**
- - Ajouter un gestionnaire d'événements `click` pour chaque carte :
- - Ignorer si le plateau est verrouillé ou si la carte est déjà retournée.
- - Changer la couleur de la carte pour celle définie dans `data-color`.
- - Ajouter la classe `.flipped`.
- Etape 5
- **Logique des paires**
- - Si c'est la première carte :
- - La stocker dans `firstCard`.
- - Si c'est la deuxième carte :
- - La stocker dans `secondCard`.
- - Vérifier si les deux cartes ont la même couleur (`data-color`).
- Etape 6
- **Gestion des correspondances**
- - Si les couleurs correspondent :
- - Ajouter la classe `.matched` aux deux cartes.
- - Réinitialiser `firstCard` et `secondCard`.
- - Si les couleurs ne correspondent pas :
- - Verrouiller le plateau pour empêcher d'autres interactions.
- - Après un délai (1 seconde) :
- - Réinitialiser les cartes en leur redonnant leur couleur initiale (grise) et en retirant la classe `.flipped`.
- - Réinitialiser `firstCard`, `secondCard` et déverrouiller le plateau.
- Bonus
- ### Points supplémentaires
- 1. Disposer les cartes aléatoirement
- 2. Ajout d'un score
- 3. Fin de partie

💬 Message dans le chat

- Objectif : variables, constantes, interpolation, .length, conditions.
- 🟢 Niveau 1 – Compteur simple
- Demande à l'utilisateur un message via prompt.
- Affiche : "Ton message fait X caractères."
- 🟡 Niveau 2 – Longueur maximale
- Si le message fait plus de 200 caractères, affiche : "Ton message est trop long pour être envoyé."
- Sinon : "Message accepté."
- 🔴 Niveau 3 – Censure d'un mot banni
- Crée une fonction filtrerMessage(message) qui :
- Remplace toutes les variantes de "nul" par "génial".
- Variantes à gérer : "nul", "nulle", "NUL", "Nulle", "NULLE", etc. (Ignore la casse et le genre.)
- 🔴 🔴 Niveau 4 – Message filtré proprement
- Affiche le message corrigé et le nombre de caractères après remplacement.
- Si aucune insulte n'est trouvée, affiche "Message propre ✅".

🔐 Validation de mot de passe

- Objectif : variables, constantes, conditions, .length, fonctions.
- 🟢 Niveau 1
- Crée une constante `motDePasseAttendu = "Coucou123"` et une variable `saisie` (en dur). Affiche si elles sont identiques ou non.
- 🟡 Niveau 2
- Si le mot de passe est correct, affiche "Bienvenue !". Sinon, affiche "Mot de passe incorrect".
- 🔴 Niveau 3
- Crée une fonction `verifierMotDePasse(mdp)` qui :
- Vérifie qu'il contient au moins 8 caractères.
- Vérifie qu'il contient "123" (simule un critère de complexité simple).
- 🔴 🔴 Niveau 4
- Si le mot de passe est trop court ou ne contient pas "123", affiche un message adapté. Sinon : "Mot de passe accepté".

🧑‍💼 Fiche profil

- 🎯 Objectifs : concaténation, interpolation, conditions, fonctions, et DOM.
- 🟢 Niveau 1 – Message simple
- Crée les variables prenom et age, puis affiche dans la console :
- "Bonjour, je m'appelle prénom et j'ai âge ans."
- Utilise l'interpolation (pas de concaténation ici).
- 🟡 Niveau 2 – Message conditionnel
- Si age est supérieur ou égal à 18, ajoute à ton message :
- "Je suis majeur(e)."
- Sinon : "Je suis mineur(e)."
- Affiche le message complet dans la console.
- 🟠 Niveau 3 – Fonction de génération
- Regroupe ta logique dans une fonction genererProfil(prenom, nom, age)
- Elle retourne une chaîne de caractères personnalisée comme :
- "Bonjour, je m'appelle Jessica Soreau et j'ai 34 ans. Je suis majeure."
- 🔵 Niveau 4 – Affichage sur la page (DOM)
- Crée une section <section id="profil"></section> dans ton HTML.
- Utilise JavaScript pour ajouter un paragraphe dans cette section, contenant le message généré.
- 🔴 Niveau 5 – Carte profil stylisée avec image
- Crée une div "carte profil" contenant :
- Une image de profil (URL : https://placehold.co/100)
- Le message retourné par ta fonction
- Ajoute cette carte à la section #profil, avec un peu de mise en forme en JS :
- bordure, ombre, fond clair, texte centré, image arrondie
- 🔴 🔴 Niveau 6 – Bonus facultatif
- Propose une fonction afficherProfil(prenom, nom, age) qui :
- Génère la carte complète
- Et l'ajoute dynamiquement au DOM
- (Tu peux ensuite l'appeler plusieurs fois avec différents prénoms)

🧑‍💼 Gestionnaire de tâches

- 🎯 Objectifs : Manipulation de tableaux, fonctions, boucles, conditions et DOM.
- 🟢 Niveau 1 – Créer une tâche et la mettre dans un tableau
- Crée un champ de saisie et un bouton "Ajouter". Lorsqu'on clique sur "Ajouter", la tâche saisie doit être ajoutée à un tableau `tasks`.
- Affiche un message dans la console pour vérifier que la tâche a bien été ajoutée au tableau.
- 🟡 Niveau 2 – Afficher les tâches sous forme de cartes
- Pour chaque tâche dans le tableau `tasks`, crée une carte HTML affichant le texte de la tâche.
- Utilise une boucle pour afficher dynamiquement toutes les tâches dans un conteneur `<div id="taskList"></div>`.
- 🟠 Niveau 3 – Supprimer une tâche
- Ajoute un bouton "Supprimer" à chaque carte de tâche. Lorsqu'on clique sur ce bouton, la tâche correspondante doit être supprimée du tableau `tasks`.
- Mets à jour l'affichage des tâches après la suppression.
- 🔵 Niveau 4 – Modifier une carte
- Ajoute un bouton "Modifier" à chaque carte de tâche. Lorsqu'on clique sur ce bouton, ouvre un `prompt()` pour permettre à l'utilisateur de modifier le texte de la tâche.
- Si un texte valide est saisi, remplace la tâche dans le tableau et mets à jour l'affichage.
- 🔴 Niveau 5 – Bonus facultatif
- Propose une fonction `editTask(index)` qui permet d'ouvrir un prompt avec le texte actuel de la tâche. L'utilisateur peut modifier la tâche et valider.
- La tâche modifiée est ensuite mise à jour dans le tableau et l'affichage est réactualisé sans recharger la page.

🛒 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.

## Snake Quiz – QCM Dynamique avec Timer

- 💬 Exercice – Snake Quizz
- Objectif : tableaux, conditions, fonctions, événements, setInterval, clearInterval.
- 🟢 Niveau 1 – Une seule question
- Crée un objet question avec :
- un intitulé
- un tableau de 3 réponses
- l'index de la bonne réponse
- Affiche la question dans la page avec 3 boutons de réponse.
- Quand l'utilisateur clique sur un bouton, affiche dans la console :
- "Bonne réponse !" ou "Mauvaise réponse."
- 🟡 Niveau 2 – Plusieurs questions
- Place plusieurs questions dans un tableau nommé quiz.
- À chaque réponse, passe à la question suivante.
- Quand il n'y a plus de questions, affiche : "Fin du quiz ! Score : X"
- (avec X le nombre de bonnes réponses).
- 🔴 Niveau 3 – Minuteur par question
- Ajoute un compte à rebours de 10 secondes pour chaque question.
- Si le temps est écoulé, passe à la question suivante et considère la réponse comme fausse.
- Utilise setInterval() et clearInterval() pour gérer le minuteur.
- 🔴 🔴 Niveau 4 – Vies et redémarrage
- Ajoute une variable vies = 3.
- Si une mauvaise réponse est donnée (ou si le temps est écoulé), enlève une vie.
- Si vies === 0, affiche "Game Over" et arrête le quiz.
- Ajoute un bouton "Rejouer" pour recommencer le quiz depuis le début.

## Rendu final JS – Génération d'un site web dynamique à partir d'une API

- 💻 📝 Rendu final JS – Génération d'un site web dynamique à partir des données d'une API
- Objectif : utilisation de fetch() pour récupérer des données JSON, manipulation DOM, boucles, gestion d'images et cartes interactives.
- 🟢 Niveau 1 – Récupérer les données de l'API
- Utilise fetch() pour récupérer les données JSON depuis l'API https://js-dynamic-portfolio-data-makerslab-emlyon-cdweb-8f83155c64a0cc.gitlab.io/.
- Tu as le choix du sujet, donc clique bien sur le sujet de ton choix pour avoir l'URL finale.
- Vérifie si la réponse de l'API est correcte. Si oui, passe les données à une fonction qui affichera le contenu du site.
- 🟡 Niveau 2 – Affichage du contenu principal
- Dans la fonction, crée un élément div pour afficher le nom de l'entreprise, sa phrase d'accroche et un bouton d'appel à l'action.
- Affiche ces informations dynamiquement avec les données récupérées depuis l'API.
- 🔴 Niveau 3 – Afficher les activités
- Pour chaque activité dans les données JSON, crée dynamiquement une div contenant un titre, une description et une image.
- Affiche ces informations dans une section dédiée, avec une carte pour chaque activité si les données contiennent un lien vers une image.
- 🔴 🔴 Niveau 4 – Afficher les témoignages
- Pour chaque témoignage dans les données JSON, crée dynamiquement une div contenant le prénom, le commentaire.
- Ajoute ces témoignages sous les activités dans la page.
- 🔴 🔴 🔴 Niveau 5 (optionnel) – Ajouter une carte interactive
- Ajoute une carte interactive à la page à l'aide de la bibliothèque Leaflet.js.
- Utilise les coordonnées fournies dans les données JSON pour centrer la carte et y ajouter un fond de carte interactif.
- 💎 Bonus – Design et personnalisation
- Fonts personnalisées : Ajoute des fonts depuis Google Fonts
- Icônes : Utilise une bibliothèque comme Font Awesome ou Material Icons
- Favicon personnalisé : via balise <link rel="icon" ...>
- Design : styles CSS avec couleurs, espaces, ombres, transitions, animations légères
  ```