Script client

Méthodologie des exercices

Exercice 1 : Le taquin

- Récupération des éléments html (querySelector)
- Recherche de la variable meilleur temps dans le localstorage, si elle existe je l'affiche
- Ajout des fonctions chrono (démarrer le chrono, le mettre en pause et le remettre à zéro) , et de la fonction qui permet d'ajouter un 0 devant les secondes, minutes et heures inférieures à 10
- Au chargement de la page, le chrono démarre, l'ordre des cartes du jeu est mélangé et le jeu est construit en fonction de l'ordre mélangé
  - Jusqu'à ce que le nombre de ligne ne soit égal à 4 et que pour chaque ligne, tant que le nombre de colonnes n'est pas égal à 4, je crée une case auquel on assigne un id qui sera le numéro de ligne et de colonne séparé d'un tiret.
  - ensuite pour chaque case on donnera la source image, images/nb.jpg => nb étant la première cellule du tableau ordre obtenu plus haut.
  - Ajout des écoutes des événements pour ces cases nécessaire au déplacement plus tard
- Appel des fonction du chrono, lors du clic sur le bouton pause
- Appel de la fonction mix, lors du click sur le bouton mix
- Initialisation des fonctions nécessaire au déplacement drag and drop des cases
- Si la case target (case blanche) n'est pas la case portant l'image 16 -> pas d'action
- Récupération de l'id de la case qui correspond à l'emplacement de celle-ci, pareil pour la case target sur laquelle on glisse la case cliquée, et comparaison pour connaître si elles sont adjacentes, si elle le sont on échange leur position et la source de leur images
- Contrôle de la position de chaque case (en fonction de la src image de chaque case) pour voir si la grille est dans l'ordre, si elle est dans le bon ordre, je stoppe le chrono et compare avec le meilleur temps stocké dans le localstorage si il y en a un pour voir si celui si est meilleur, je stocke le plus petit temps des deux.

Exercice 2 : Le memory

- Récupération des éléments de l'HTML (querySelector)
- Recherche d'une valeur meilleur temps dans le localStorage si elle existe et affichage du meilleur temps si il existe
- Ajout des fonctions chrono (démarrer le chrono, le mettre en pause et le remettre à zéro) , et de la fonction qui permet d'ajouter un 0 devant les secondes, minutes et heures inférieures à 10
- Ajout d'une fonction pour recommencer le jeu, en rechargeant la page
- Ajout d'une fonction qui mélange les cases en échangeant les positions des valeurs du tableau
- Ajout d'une fonction construct qui construit le jeu: tant que i n'est pas égal à 20 , je crée une case, et lui donne un attribut data qui sera son numéro de case et un attribut vue qui décidera si on affiche le dos de la carte ou non, pour l'instant toutes les cartes possède l'attribut vue de valeur false, et on donne la src de l'image qui correspond au dos d'une carte contenu dans le dossier image.
- Au chargement de la page, appel de la fonction qui mélange le jeu, puis celle qui construit le jeu. Setting d'un tableau de choix vide, qui se remplira à la sélection de

deux cartes pour pouvoir comparer les deux valeurs de ce tableau afin de savoir si les deux cartes présentent la même image.
- Lorsque je clique sur une case, le chrono se met en route, et la src de l'image de la case est settée pour que l'image de la carte s'affiche et reste affichée jusqu'à choisir une deuxième carte. Je récupère la valeur de l'attribut data des deux classes cliquée de j'insère dans un tableau pour comparer les deux valeurs. Si elle sont égales, le nombre de paires trouvées augmente de 1, et l'attribut vue, passe à true, pour rester retournée.
- Si les deux cartes sélectionnées ne présente pas la même image, les deux cartes reprennent la src de leur image comme l'image correspondant au dos de carte.
- Une fois 10 paires trouvées, le chrono s'arrête et stocke le meilleur temps dans le localstorage, puis le jeu redémarre.

Exercice 3 : Le convertisseur

- Récupération des éléments de l'HTML (querySelector)
- Collecte des infos contenu dans l'url fourni grâce à fetch
- Calcul du résultat dès chargement de la page avec les valeurs sélectionnées dans le formulaire
- Ajout d'un addEventListener sur chaque input (monnaie d'origine et monnaie de conversion) pour « écouter les changements et calculer directement en conséquence le nouveau montant qui serra affiché sous le formulaire, et stockage de la monnaie d'origine et de conversion dans le localstorage pour que lorsque l'utilisateur revienne sur ce site, les dernières valeurs utilisées soit déjà par défaut dans le formulaire.
- Si le montant d'origine à convertir n'est pas un nombre, l'application considèrera le montant d'origine comme 1.
- Méthodologie équivalente pour le convertisseur de bitcoin

Exercice 4 : Le lotto

- Récupération des éléments de l'HTML (querySelector)
- Recherche dans le localStorage s'il existe une données correspondant au portefeuille, si pas je donne une valeur par défaut de 100€
- Ensuite, je procède à l'insertion des numéros dans les cases
- Ensuite, pour chaque case, j'écoute l 'événement click, et attribue un attribut actif pour pouvoir colorer les cases sélectionnées, et les insère dans un tableau choix qui comptera 5 céllules. L'attribut data true ou false sert à la déselection des chiffres en cas de changement d'avis ou d'erreur de sélection des chiffres. Je procède de l amême manière pour les numéros complémentaires, le tableau comptera ici 2 cellules.
- Ensuite, ajout d'un addEventListener click sur le bouton pour valider les numéros choisis et étoiles choisies. On récupère la valeur de l'input mise, celle-ci devra être minimum de 10.
- Tant que le nombre de numéros attendu n'est pas atteint, j'affiche le nombre de numéros qu'il reste à cocher.
- Tirage au sort des numéros du lotto et des étoiles, et insertion dans les tableau tirage et comptirage (complémentaires/étoiles)
- J'affiche mon tirage, en créant un div pour chaque élément du tableau tirage pour afficher les boules dans lesquelles seront affichées mes numéros. Technique équivalente pour l'affichage du tirage officiel du lotto.
- Contrôle des numéro en comparant chaque nombre du tableau de mon tirage à ceux du tirage lotto , pareillement pour les complémentaires.

- Calcul des gains en fonction du nombre de numéros et de complémentaires égaux à la suite de la comparaison des tableaux. Affichage des gains et recalcul et stockage du portefeuille dans le localStorage.

Exercice 5 : Le formulaire de commande

- Récupération des éléments de l'HTML (querySelector)
- Ajout d'un addEventListener click sur le bouton plus, qui insère un bloc comprend trois champ input destiné à entrer la quantité, le produit et le prix, du produit que l'on veut ajouter à la commande. Ajout d'un compteur de produits ajouté, à chasue click sur le bouton plus.
- Au click sur le bouton commander, récupération des valeurs entrées dans le formulaire pour les coordonnées et les produits,
- Calcul de la somme du ticket grâce à la récupération des données entrées dans les champs prix et quantité.
- Contrôle du formulaire, si les champs ne sont pas remplis, message d'erreur.
- Si les champs sont bien remplis, enregistrement des coordonnées dans un tableau dans le localStorage
- Envoi des data via une requête XMLHttpRequest sur un fichier response.php
- Affichage du tickets reprennant les coordonnées client, les produits commandés et la somme du ticket à payer

Exercice 6 : Le blackjack

- Récupération des éléments de l'HTML (querySelector)
- Récupération de la valeur portefeuille dans le localStorage si elle existe, et si elle n'existe pas, définition d'une valeur par défaut
- Ecoute des changement lors de l'entrée d'une valeur dans le champ mise, récupération de cette valeur et construction du jeu, mélange des cartes et fonction jouer()
- La fonction contstruct du jeu, établit le jeu de cartes selon un tableau de valeur de 1 à 10 plus valet, roi, dame et un tableau définissant le type de carte (coeur, pic, carreaux, trèfle) en associant chaque élément du premier tableau avec les éléments du deuxième en les insérant dans un troisième tableau ('jeu')
- La function mix() mélange le jeu en échangeant les position des valeurs du tableau 'jeu' sortit de la function construct
- La fonction jouer :
  - attribue la carte cachée de l'ordinateur comme la dernière carte du jeu mélangé
  - calcule la valeur de la carte cachée de l'ordinateur
  - contrôle si la carte cachée est un as
  - tant que la somme des cartes de l'ordinateur n'excède pas 17 l'ordinateur pioche des cartes, pour cela, on crée des éléments, auquel on attribue une img src correspondant à son nom que l'on obtient en récupérant la dernière carte du jeu, et en recalculant à chaque fois la somme des cartes de l'ordinateur et en contrôlant si la carte est un as pour lui attribuer sa bonne valeur
  - distribution de deux cartes pour le joueur, procède de la même manière pour afficher les cartes. Calcul de la somme de mes cartes, vérification des as.
- Ajout d'un AddEventListener sur les boutons pioche et tapis.
  - Le bouton pioche appelle la fonction pioche qui affiche une carte, la supprime du jeu et calcule la somme de toutes mes cartes en vérifiant les as
  - Le bouton tapis appelle la function stay et retourne la carte cachée de l'ordinateur, et compare la somme de la valeur des cartes de l'ordinateur et la somme de la valeur des cartes du joueur et affiche le message de gain ou de perte en fonction du

résultat obtenu. Recalcul du portefeuille en fonction de la mise et stockage de la valeur dans le localstorage.
- La fonction recommence, relance une partie à partir d'un certain moment après l'affichage du message de gain ou de perte.

Exercice 7 : Le morpion

- Récupération des éléments de l'HTML (querySelector)
- Création d'une fonction vider() qui vide le plateau de jeu
- Création d'une fonction insert() qui nous permet de récupérer la valeur dans les cases pour chaque cases des 3 lignes et de les insérer dans un tableau
- Ensuite pour chaque cases j'écoute l'évènement click, si le coups à jouer est pair j'insère un O si il n'est pas pair j'insère un X dans la case, je change la phrase qui indique quel joueur doit cocher une case.
- Attribution des points en fonction de la ligne complétée par des X ou des O et empêchement du click sur les cases une fois le point gagné. Affichage des points. Vidange du tableau.
- Ecoute du click sur le bouton rejouer pour réactiver le click sur le jeu et relancer une partie.
- Ecoute du click sur le bouton restart, celui-ci recommence la partie en remettant le compteur de points à 0 pour les deux joueurs.

Exercice 8 : La transformation d'image

- Récupération des données de l'HTML (querySelector)
- Au chargement de la page, récupération des données du localstorage et si elles existent, attribution de celles-ci à l'image
- Ecoute du scroll sur l'image, au scroll j'augmente ou diminue la largeur et la hauteur de la photo, en fonction du scroll vers le haut ou le bas et que la largeur de la photo est au dessus de 200px. + stockage de la nouvelle valeur dans le localstorage
- Pour la rotation de la photo, mise en place de deux bouton fléchés qui ont chacun la fonction de faire tourner l'image, soit dans le sens horloger soit dans le sens contraire du sens horloger. Pour cela, ajout d'écoute du click sur ces boutons et calcul du degré de rotation à appliquer à l'image. + stockage de la nouvelle valeur dans le localstorage
- Pour le déplacement en mode drag and drop de l'image, calcul de la position du curseur dans la fenêtre. Déplacement de l'image où on lâche le clic, en appliquant à l'image (margin-top et left) les valeurs de la position de la souris.

Exercice 9 : Le parallax

- Récupération des éléments de l'HTML (querySelector)
- Définition de la zone de parallax et définition de la longueur et la largeur de chaque plans du parallax dans la css
- (HTML) Définition du data-x et du data-y comme leur position dans l'espace de la zone parallax
- (HTML) Définition du data-move-x et du data-move-y comme la manière dont ils vont réagir au mouvement de la souris, de quelle proportion sera leur mouvement vertical et horizontal lors de l'évènement défini dans le script
- Au mouvement de la souris, attribuer aux plans du parallax les données de l'emplacement de la souris
- Calcule des mouvements des différentes couches.
- Pour l'oiseau, écoute de l'évènement over et attribution de nouvelles positions et setting de la transition d'animation à 4s