Jeu du pendu

Il va vous falloir développer un jeu de pendu. L'utilisateur se verra indiquer le nombre de caractère d'un mot secret.
par exemple pour un mot de 4 lettres "----". Il devra alors choisir des lettres une à une pour tenter de découvrir le mot complet. Il faut indique à l'utilisateur les lettres qu'il a déjà utilisé. Si il trouve une lettre du mot, il faut afficher les lettres par exemple si il trouve la lettre "e" "e--e" Si il se trompe trop de fois (6 ou 7 fois par exemple), il perdra alors la partie. Il faut indiquer à l'utilisateur où il en est de ses échecs. Le nom du jeu étant lié à la représentation d'un bonhomme bâton sur une potence que l'on dessine petit à petit à chaque échec jusqu'à ce qu'il soit totalement dessiné, on peut garder ce thème ou bien le changer.

PARTIE 1

Côté HTML il vous faudra : Afficher le nombre d'erreur. (Faites simple pour commencer, un chiffre sera suffisant.) Afficher les lettres trouvés. Afficher les lettres selectionnable. Côté CSS : Vous êtes libre sur le design, mais tentez de faire quelque chose de responsive. Côté JS : Il faudra que vous ayez un mot à deviner pour votre pendu. Commencez avec un seul mot dans une variable. Il faudra afficher à l'utilisateur le nombre de lettre que contient votre mot. Il vous faudra que tous vos boutons réagissent au clique. Faites des consoles log pour voir si vous arrivez bien à obtenir la lettre correspondant à votre bouton. Une fois la lettre du bouton obtenu, il faudra vérifier si elle est présente dans votre mot. Si Oui, alors il faudra afficher à l'utilisateur où elle se trouve dans votre mot. (Une fois votre affichage réussi, il faudra penser que certains mots peuvent contenir plusieurs fois la même lettre.) Si Non, il faudra indiquer à l'utilisateur qu'il s'est trompé et se rapproche de la fin. Si l'utilisateur trouve toute les lettres, il faudra lui indiquer qu'il a gagné. Si le nombre de tour maximum est atteint, il faudra lui indiquer qu'il a perdu.

PARTIE 2

Remplacer le mot unique, par un tableau contenant plusieurs mots et en choisir un au hasard. Remplacer le compteur d'échec par un dessin ou une animation qui change à chaque échec. Permettre à l'utilisateur de pouvoir appuyer sur les touches du clavier pour valider les lettres. Il serait agréable pour l'utilisateur qu'il ne puisse pas sélectionner à nouveau une lettre déjà utilisé. Lors de la défaite ou de la victoire, afficher quel était le mot. Proposer au joueur de relancer une partie sans qu'il ai besoin de recharger la page manuellement.

PARTIE 3

Remplacer le tableau de mot par une requête sur un fichier JSON (vous pouvez créer le votre ou utiliser "mots.json") Lorsque le mot est affiché en fin de partie, il serait bien qu'il soit cliquable et redirige vers la recherche google du mot. On pourrait ajouter un choix de difficulté en début de partie qui en mode facile afficherait la première lettre du mot.

PARTIE 4

Utiliser le second fichier JSON (themes.json), sa structure étant plus complexe il faudra : Choisir un thème aléatoire, Une lettre aléatoire. Un mot aléatoire. Cette liste contenant énormément de mot, il faudra vérifier que le mot sélectionner ne contienne que des lettres qui correspondent à vos boutons. (Si le mot sélectionné contient le caractère "â" ou "-" mais que vos boutons ne le gère pas, alors il faudra au choix : recommencer la

recherche de mot, offrir ces lettres à l'utilisateur ou adapter vos touches pour que ces lettres soient gérées.) Cette liste contient aussi pour certains thèmes des lettres sans mots, donc attention que votre code gère cette possibilité. Afficher le thème correspondant au mot choisi. Ajouter un mode de difficulté difficile qui cache le thème du mot.