582-32F-MA - Programmation d’interface Web 2

TP1

Description

Produire une interface dynamique à l’aide de JavaScript utilisant la programmation orientée objet modulaire.

Énoncé

Vous avez obtenu le contrat pour la refonte du site d’une librairie en ligne. Pour le premier sprint, votre équipe vous a attribué le développement de l'interface côté-client listant les livres en inventaire. Celle-ci affiche les livres sous différentes catégories ainsi que le détail d’un livre suite au clic de sa tuile.

Vous n'avez pas encore accès aux données du serveur, mais les développeurs backend vous ont fourni un fichier JSON contenant les données des livres ainsi que toutes les images nécessaires. L'affichage des livres doit se faire dynamiquement à partir de ce fichier JSON sans que PHP ou autre langage côté serveur soit utilisé, c'est-à-dire que vous devez utiliser uniquement HTML, CSS et JavaScript pour accomplir cette tâche.

Votre chef de projet vous a imposé d'utiliser la programmation orientée objet pour structurer votre code JavaScript. Vous devez créer des classes pour représenter les livres, les catégories et la boîte modale. Chaque classe doit avoir des méthodes pour gérer les interactions utilisateur, telles que le filtrage des livres par catégorie et l'affichage des détails d'un livre dans une boîte modale.

Objectifs

- Utiliser la programmation orientée objet pour structurer le code d'une interface front-end
- Manipuler des données statiques en format JSON
- Écrire un code propre, bien structuré sans bug et sans avertissement

Consignes

Sources de données

- L’application backend n’est pas complète, du coup vous devrez travailler avec des données statiques en format JSON (fourni).
- Toutes les images sont également fournies. Je vous invite à les placer suivant l’arborescence `./assets/img/`, autrement il vous faudra changer tous les chemins relatifs du fichier livres.js car c'est ainsi que les chemins ont été enregistrés.

Affichage des tuiles

- Afficher, selon la maquette, la liste des livres.
- Chaque tuile d’un livre doit afficher son image, son titre, son prix et un bouton pour l’ajouter au panier (celui-ci n'est pas fonctionnel pour l'instant).

Filtres

- Au chargement, les nouveautés sont affichées.
- Il y a 8 filtres, ceux-ci sont :
    - Tous
    - Nouveautés
    - Littérature
    - Art de vivre
    - BD, Jeunesse, Humour
    - Culture et société
    - Loisirs, Tourisme, Nature
    - Savoir et science
- Remarquez que `Nouveauté` n'est pas une catégorie mais bien une clé booléene présente pour chaque livre. Bref, au clic de Nouveauté, vous devez afficher tous livres où sa clé nouveaute est `true`.
- Évitez la redondance et faites une seule fonction qui gère tous les filtres.
- Indiquez visuellement le filtre actif.

Boite modale

- Au clic d'une tuile (toute la tuile doit être cliquable), une boite modale affiche les informations complète de ce livre, soit : son image, son titre, son auteur, son éditeur, son nombre de pages et sa description. La boite modale doit être centrée à l'écran et le reste de la page doit être assombri.
- Empêchez le scroll vertical de la page lorsque la boite modale est ouverte (N'oubliez pas de remettre le comportement correctement à la fermeture de celle-ci).
- Une fois la boite modale ouverte, l'usager peut la fermer en cliquant n'importe où sur celle-ci. Prenez tout de même la peine de placer un X en haut à droite pour le design.

Mise en ligne

- Vous devez utiliser Github Pages pour mettre en ligne votre site. Pour ce faire, vous devez activer Github Pages dans les paramètres de votre dépôt.
- Vous devez fournir le lien vers votre site dans le dossier `.Zip` remis dans un fichier README.md.

Stratégies de développement

Ce TP met à l'épreuve votre capacité à utiliser la programmation orientée objet. Démontrez votre compréhension des concepts de base de la POO en structurant votre code de manière modulaire et en utilisant les concepts de classes, d'héritage, d'encapsulation et de polymorphisme.

Aucun script procédural n'est accepté : le fichier main.js ne sert qu'à lancer les instances de classes nécessaires au chargement de la page et chaque bloc fonctionnel devrait avoir sa classe.

Vous aurez minimalement besoin d'une classe Livre, d'une classe LivreModale et d'une classe Filtre mais vous pouvez ajouter d'autres classes si vous le jugez nécessaire.

Design

Vous avez une grande liberté pour le design. Vous devez cependant respecter la maquette fournie au niveau des fonctionnalités. Vous pouvez utiliser des polices Google Fonts, des icônes FontAwesome, etc. Soyez créatif tout en restant professionnel. Ne prenez pas la peine de faire un design responsive pour ce TP. L'aspect programmation de l'interface est le plus important.

Astuces

1. Commencez par écrire le fichier de classes vides, puis ajoutez les méthodes et propriétés nécessaires une à une.
2. Séparez le code en petites fonctions pour faciliter la lecture et la maintenance. Ex: une fonction pour afficher les tuiles, une autre pour gérer les filtres, etc.
3. Utilisez des noms de propriétés et de méthodes claires. Pas de noms abrégés ou obscurs.
4. Testez au fur et à mesure que vous avancez.
5. Dans l'ordre, faites afficher les tuiles, puis les filtres, puis le modal.

Modalités de remise

Date de remise

Le site doit être en ligne au plus tard le **6 octobre 23h59**, avant le cours 10.

Remise

Pour démontrer l'évolution de votre projet, vous devez effectuer des commits réguliers. Vous devriez en faire un minimum de 10.

Vous devez me remettre un dossier Zip de votre projet sur Teams dans la section `devoirs` fonctionnel avant la date de remise. Le dossier doit contenir un fichier README.md avec le lien vers votre site Github Pages.

Pondération

Le travail compte pour 20% de la note finale

Retard

Selon les règles du collège, 5% par jour de retard seront enlevés, jusqu'à 5 jours de retard maximum.

Critères d'évaluation

- Réussite des différentes fonctionnalités et respect des consignes
- Utilisation efficace de la programmation orientée objet (classes, modules, héritage, encapsulation, polymorphisme)
- Qualité des algorithmes et qualité du code source et des tests.
- Le code ne contient aucun avertissement, aucune erreur et aucun console.log au moment de l'exécution
- Structure et optimisation du code (indentation, nommage, commentaires, structure des dossiers, etc.)
- Autonomie et professionnalisme: Présence régulière de commits et le projet a été présenté à l'enseignant 1 fois avant la remise
- Le site est en ligne sur Github Pages

Plagiat

Vous devez citer les extraits de code qui dépassent une ligne ou deux et suivre un tutoriel (en tout ou en partie) sera considéré comme du plagiat. Vous avez tout dans les notes de cours pour réussir ce travail. `L'utilisation d'un générateur de code par intelligence artificielle est interdite`. Utilisez les techniques de programmations vues en classe.

Il s'agit d'un travail strictement individuel. Vous pouvez poser des questions à vos collègues, mais ce doit être votre logique et vos scripts. L'objectif est de développer votre autonomie. En cas de plagiat, je devrai appliquer les sanctions de la PIEA (Politique institutionnelle d'évaluation des apprentissages).

Si vous êtes vraiment bloqué, n'hésitez pas à m'écrire sur Teams. N'attendez pas à la dernière minute pour demander de l'aide.