programation web en js

TD n°6 : fetch et promesses, photobox

Préambule

l'objectif du TD est de comprendre la notion de promesse et son utilisation dans des requêtes ajax asynchrones réalisées avec l'api javascript fetch.

Dans les exercices suivants, on utilise une api disponible sur webetu, dont le point d'entrée est : https://webetu.iutnc.univ-lorraine.fr/www/canals5/phox/api

et la documentation complète disponible ici : https://webetu.iutnc.univ-lorraine.fr/www/canals5/phox/doc/

Cette api met à disposition des galeries d'image qu'il est possible de parcourir et d'afficher.
Pour explorer cette api et notamment comprendre la structure du json retourné dans les réponses, il est conseillé d'installer une application dédiée, par exemple :

- bruno : https://www.usebruno.com/
- postman : https://www.postman.com/
- insomnia : https://insomnia.rest/

Pour accéder à cette api installée sur webetu, une authentification http est nécessaire pour toute requête en provenance d'un réseau extérieur à l'IUT. Pour tenir compte de cette contrainte, vous avez 2 possibilités :

- soit, utiliser le VPN @etu : documentation disponible ici : https://wikidocs.univ-lorraine.fr/pages/viewpage.action?pageId=46892370 ou accessible par ici : https://wikidocs.univ-lorraine.fr/display/dociutcharlemagne/Accueil, rubrique "Département Informatique".
- ou bien, ajouter dans toutes vos requêtes fetch l'option `{ credentials : 'include' } :`

```
fetch(https://webetu.iutnc.univ-lorraine.fr/www/canals5/phox/api/photos/148,
      { credentials: 'include'}
)
```

Exercice 1 : afficher une image

Etape 1 : récupérer les données d'une image

On souhaite programmer une fonction permettant de récupérer les données correspondant à une photo dont l'identifiant est passé en paramètre et de les afficher dans la console.
Pour cela :

a) créer le module `photoloader` qui regroupe toutes les fonctionnalités d'interaction avec l'api photobox. Dans un premier temps, le module exporte une fonction `loadPicture( idPicture )` ; la fonction reçoit un identifiant d'image et retourne une promesse qui se résout en un objet javascript contenant la description de la photo. Les erreurs sont traitées dans cette fonction.
b) les valeurs de configuration éventuelles, par exemple le point d'entrée de l'api (`"https://webetu.iutnc.univ-lorraine.fr/www/canals5/phox/api"`) sont à déclarer dans un module séparé importé par le module photoloader.
c) créer le module `index` chargé par `index.html`. Ce module importe le module `photoloader`, et déclare la fonction `getPicture( id )` qui reçoit un identifiant de photo en paramètre et

affiche dans la console le titre, le type et l'url de cette photo. Utiliser cette fonction pour la photo 105, puis récupérer le numéro de photo dans l'url :
`index.html#106 → window.location.hash`

Etape 2 : afficher l'image
On souhaite maintenant afficher l'image et les données qui l'accompagnent dans la page. Utiliser pour cela le code html fourni dans le squelette.

a) créer le module `ui` ; dans ce module, définir et exporter la fonction `displayPicture` qui reçoit un objet javascript correspondant à une image, et insère cette image avec les données descriptives dans le document html, à l'endroit prévu (`#la_photo`). Le fragment html inséré est construit à l'aide d'un template Handlebars.
b) importer ce module dans le module index et utiliser cette fonction `displayPicture` pour afficher les données d'une image dans le document plutôt que dans la console.

Etape 3 : requêtes complémentaires pour la catégorie et les commentaires de l'image
On souhaite maintenant compléter l'affichage de l'image en ajoutant le nom de la catégorie à laquelle elle appartient et les commentaires associés à cette image.
Pour obtenir ces informations, des requêtes supplémentaires auprès de l'api sont nécessaires. On utilise pour cela les URI qui sont retournées dans les réponses, dans la propriété `links.`

a) pour une photo, examiner les données retournées par l'api dans la partie links et utiliser l'uri permettant d'accéder à la catégorie de la photo dans votre application `{bruno,postman, insomnia }` pour connaitre la structure des données décrivant la catégorie.
b) créer dans le module `photoloader` une fonction `loadResource( uri )` qui reçoit une URI, charge (fetch) la ressource correspondante et retourne une promesse qui se résout avec les données renvoyées par l'api. Exporter cette fonction.
c) dans le module `index`, créer une fonction qui reçoit les données d'une image et retourne une promesse qui se résout avec les données de sa catégorie. Elle utilise le lien "categorie" de l'image et la fonction créé au b).
d) dans le module `ui`, créer et export une fonction qui reçoit les données de la catégorie et insère le nom de la catégorie dans l'affichage de la photo, à l'endroit prévu à cet effet (`#la_categorie`).
e) compléter la fonction `getPicture` pour qu'elle ajoute la catégorie dans l'affichage de l'image.
f) Sur le même principe, traiter les commentaires. Chaque commentaire est ajouté dans un élément `<li>` dans la liste `#les_commentaires.`

Le résultat final doit correspondre à l'image en annexe 1.

Exercice 2 : afficher 1 galerie de photos

L'objectif est d'afficher une liste de photos sous forme de vignettes. La liste de photos est obtenue grâce à une requête fetch auprès de l'api photobox.
La galerie est construite en vous basant sur le markup html et le code css fourni dans le squelette. Vous pouvez le modifier si vous le souhaitez.
Procédez par étapes :

1. Examiner la structure des données retournées par l'api lorsque l'on accède à une collection de photos avec une requête sur l'uri https://webetu.iutnc.univ-lorraine.fr/www/canals5/phox/api/photos,
2. complétez si nécessaire le module `photoloader` de d'exercice 1 pour gérer les requêtes vers l'api. En particulier, vous utiliserez la fonction loadResource() pour charger la liste de photos initiale. La méthode retourne une promesse permettant d'accéder aux données renvoyées par l'api.

3. créez un module nommé `gallery` pour le chargement et la pagination des galeries. Ce module exporte une fonction `load()` qui charge la liste de photos (en utilisant le module photoloader), stocke les données et retourne la galerie.
4. créez le module `gallery_ui` chargé de l'affichage d'une galerie. Le module exporte la fonction `display_galerie()` qui reçoit une galerie en paramètre et l'affiche en construisant le markup html correspondant, puis en l'insérant dans le DOM. Basez-vous sur le markup html fourni dans le squelette. Remarquez que la valeur de l'attribut `data-photoId` est l'identifiant de la photo correspondant à la vignette.
5. Dans le module principal de l'application, associer l'événement "click" sur le bouton de chargement d'une galerie à l'action correspondante : charger puis afficher la galerie.

Exercice 3 : naviguer dans les galeries
On souhaite compléter la navigation dans les galerie de photos en ajoutant la possibilité de parcourir les pages de photos, en cliquant sur les boutons prévus à cet effet.

1. examiner les données retournées par l'api pour la liste de photos et identifier comment, à partir de ces données, on peut obtenir la page suivante et la page précédente.
2. compléter le module `gallery` pour y ajouter les méthodes permettant de charger la page suivante/précédente de la galerie. Pour cela, lors du chargement d'une galerie, on devra stocker les informations permettant la navigation.
    1. compléter la fonction load() déjà présente pour qu'elle stocke ces informations,
    2. programmer et exporter les méthodes next() et prev() qui chargent les données de la page suivante/précédente,
3. ajouter les listener pour les boutons "next" et "prev" dans la zone de navigation de votre interface html. Ils chargent puis affichent la galerie suivante/précédente.
4. sur le même principe, ajouter les bouton 'first" et "last" pour accéder à la première/dernière page.

Voir exemple d'affichage en annexe 2.

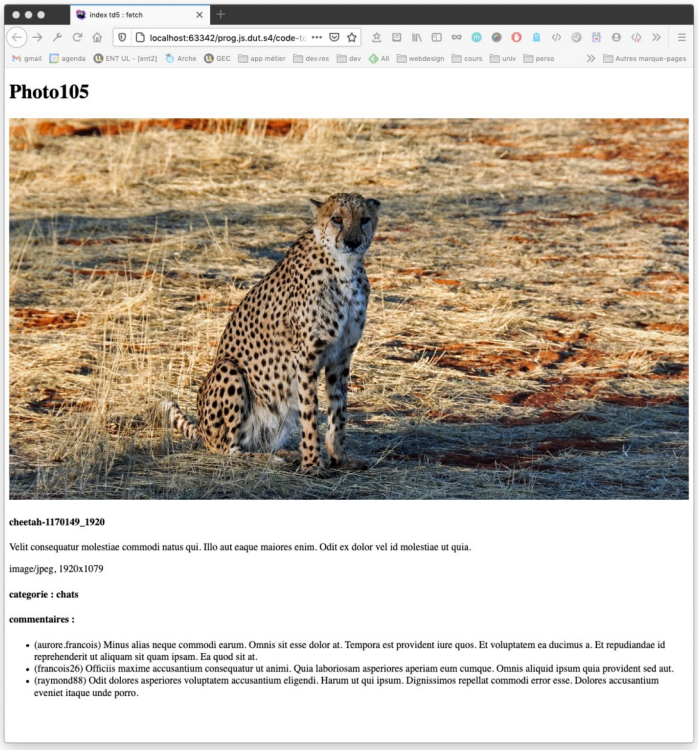
Exercice 4 : affichage d'une photo
Lorsque l'utilisateur clique sur une photo de la galerie, on affiche cette photo en format original dans une section en dessous de la galerie. Les données de la photo doivent être récupérées grâce à une requête auprès de l'api. L'uri à utiliser est `/www/canals5/phox/photos/` suivi de la valeur de l'attribut `data-photoId` de la vignette qui a été sélectionnée.

1. En réutilisant le code de l'exercice 1, compléter la méthode d'affichage d'une galerie pour qu'un clic sur une image de la galerie déclenche l'affichage de la photo dans la section `#la_photo`.

Exercice 5 (bonus facultatif) : lightbox
Transformer l'affichage d'une image (exercice 4) pour utiliser un mode *lightbox*, c'est à dire un mode où l'image occupe toute la fenêtre du navigateur et recouvre la galerie qui n'est plus visible.
Ajouter dans cette lightbox :

- un bouton pour fermer la lightbox et revenir à l'affichage de la galerie,
- des boutons pour naviguer dans la lightbox : passer à l'image suivante ou précédente sans revenir à la galerie.

ANNEXE 1 : exemple d'affichage correspondant à l'exercice 1



ANNEXE 2 : exemple d'affichage correspondant à l'exercice 3