Programmation Web « Client Riche »
R410 - TD5

Objectifs

Ce TD permet de se familiariser avec l'utilisation de jQuery. Il s'agit d'un Framework **gratuit** et **open-source** très utilisé créé pour simplifier l'utilisation de JavaScript.

En 2022, et d'après W3Techs jQuery était utilisé par 77% des 10 millions de site les plus populaires. Il est très probable que vous soyez amenés à l'utiliser au cours de votre vie, surtout si vous faites du développement web.

Il permet notamment de simplifier :

- L'accès au DOM
- Les évènements
- Les animations
- Ajax (envoi de requêtes asynchrones)

Pour ce TD, vous aurez besoin de faire des recherches sur :

- preventDefault()
- $.ajax()
- JSON.parse()

Installer jQuery

Vous pouvez inclure jQuery de deux manières différentes ; soit en utilisant un lien **CDN** soit en téléchargeant jQuery sur leur site officiel : https://jquery.com/download/

Nous allons utiliser la première méthode qui consiste à utiliser un lien de jQuery déjà hébergé afin de gagner du temps et intégrer le script en bas de page, **juste avant la fermeture du body**.

```
<script src="https://code.jquery.com/jquery-3.6.4.min.js" integrity="sha256-
oP6HI9z1XaZNBrJURtCoUT5SUnxFr8s3BzRl+cbzUq8="
crossorigin="anonymous"></script>
```

Vous pouvez retrouver d'autres versions hébergées de jQuery sur leur site : https://releases.jquery.com/

Familiarisation avec jQuery

L'utilisation de **jQuery** ressemble à celle de **querySelectorAll** à quelques exceptions près. La plus grande étant le type de retour observé. En jQuery, on utilise le dollar **$** pour aller récupérer un élément.

Pour récupérer l'élément h1, nous avons donc maintenant plusieurs possibilités :

La particularité de jQuery (ici avec **$('h1')**) c'est qu'il retourne un objet contrairement aux appels JavaScript qui retournent des éléments ou nœuds.

jQuery est livré avec de nombreuses méthodes qui simplifient la vie, voici quelques exemples :

- .html() et .text()
- .addClass() et .removeClass()
- .css()
- .remove()
- .parent() et .children()
- .on() (pour les évènements)

Par convention, on rajoute un **$** devant les noms de variables quand les variables sont des objets jQuery, mais ce n'est pas obligatoire ! (pour ceux pour qui PHP ne serait pas un bon souvenir…)

```
var $h1 = $('h1');
```

Exercice 1 : Une requête Ajax POST

Pour cet exercice, vous allez devoir relancer votre vieux serveur PHP (et oui, désolé…). Nous allons traiter le formulaire en asynchrone, en envoyant une requête à une page PHP sans quitter la page sur lequel se trouve l'utilisateur. Le fichier PHP vous est fourni dans l'archive du TD.

Le traitement de ce formulaire fera communiquer JavaScript et PHP à travers le langage **JSON**.

1) Créez un formulaire HTML contenant :

- Un champ action qui utilisera la méthode **post**
- Redirigeant vers **form.php**
- Un champ contenant uniquement le type texte et l'id « username ».
- Un champ contenant uniquement le type password.
- Un bouton de type submit avec le texte « Connexion ».

2) Avec jQuery, rajoutez un évènement **lorsque le formulaire est soumis** qui permet de :

- Stopper l'envoi du formulaire
- Récupérer les champs du formulaire
- Envoyer une requête ajax en POST avec les deux champs soumis vers l'URL du formulaire

3) Rajoutez un évènement à la saisie sur chaque champ :

- qui enlève la couleur rouge des bords du champ si au moins une lettre est saisie et la remet si le champ est vide à nouveau
- qui met les bords en rouge si le nombre de caractères saisis est > à 255

La requête Ajax devra traiter le cas d'un message d'erreur ou de succès. En cas d'erreur, le message devra être affiché en rouge au-dessus du formulaire et ne pas s'accumuler (si on renvoie le formulaire, seul le dernier message est affiché). En cas de succès, on remplace le formulaire par le message de succès en vert.

Exercice 2 : Une requête Ajax GET

Pour cet exercice, vous allez récupérer les informations d'une page externe grâce à Ajax et charger dynamiquement des faux produits en fonction d'un champ. Cela vous permettra de comprendre ce que l'on peut faire avec. Vous pourriez par exemple tenir sur votre site la météo à jour en temps réel, mais nous allons ici prendre un autre exemple.

Vous n'avez pas besoin de modifier la page HTML pour cet exercice, vous n'avez que du JavaScript à faire !

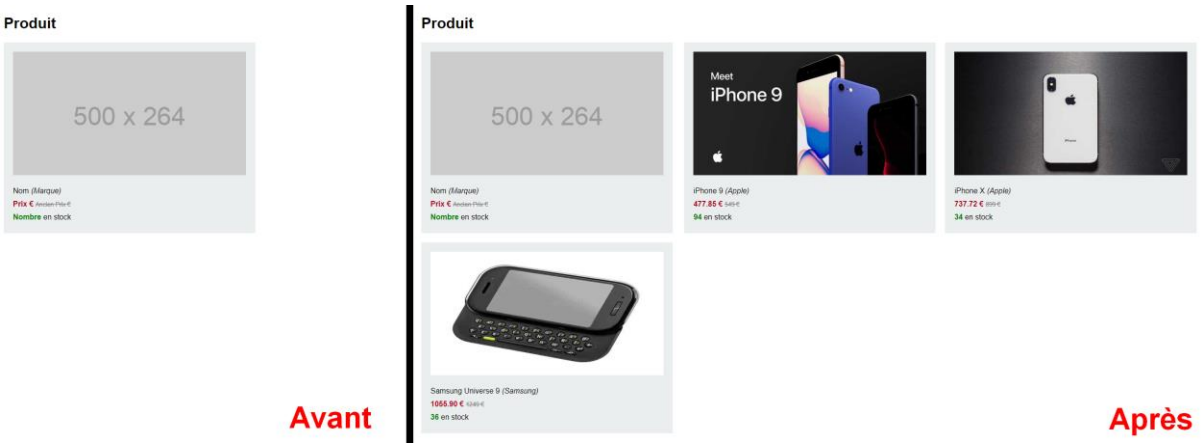
Nous allons utiliser le site https://dummyjson.com/ qui nous permet d'avoir des fausses données JSON.

1) Créez une boucle avec i allant de 1 à 3 et faisant à chaque fois une requête vers https://dummyjson.com/products/{i}

2) À chaque succès de la requête, clonez l'élément **.product-template**, supprimez la classe **.product-template**, rajoutez une classe **product-{i}** et rajoutez le à la fin du body.

3) Remplacez les données **.img** (src + alt), **.title**, **.brand**, **.price**, **.old-price** et **.stock** en utilisant le résultat JSON récupéré par la requête Ajax et en respectant la forme du template.

Voici le résultat attendu :



4) Améliorez votre code pour rajouter un évènement sur le bouton « Générer » qui génère un nouveau produit (qui n'a pas déjà été rajouté sur la page) avec un id aléatoire compris entre 1 et 100 (si les 100 produits ont déjà été générés, le bouton se mettra en « disabled »).

Exercice 3 (facultatif)

Reprenez un exercice de l'un de vos TD précédents et utilisez jQuery pour simplifier le code ! 😉