

Le format JSON et AJAX

Table des matières

I. Contexte	3
II. Le format JSON	3
A. Le format JSON	3
B. Exercice : Quiz	7
III. AJAX	9
A. AJAX	9
B. Exercice : Quiz	13
IV. Essentiel	14
V. Auto-évaluation	14
A. Exercice	14
B. Test	16
Solutions des exercices	16

I. Contexte

Durée : 1 heure

Prérequis : les bases de HTML et JavaScript

Environnement de travail : Visual Studio Code

Contexte

Lorsque nous découvrons le langage JavaScript, nous découvrons forcément deux formats utiles pour transporter des données : JSON et AJAX. Vous verrez que les deux sont étroitement liés lorsqu'on veut traiter les données.

JSON est un sous-ensemble de la syntaxe JavaScript pour transporter des données et les « *consommées* ». AJAX n'est pas un langage contrairement à JSON. Il s'agit d'une combinaison de requête (request), intégrer au navigateur, et de JavaScript et HTML DOM pour afficher ou utiliser les données.

De plus, JSON est « *autodescriptif* » et facile à comprendre lorsqu'on analyse (parse) des données. Quant à AJAX, il a cette particularité de lire les données d'un serveur web, après le chargement de la page, mettre à jour une page web sans recharger la page et envoyer des données à un serveur web en arrière-plan.

Attention

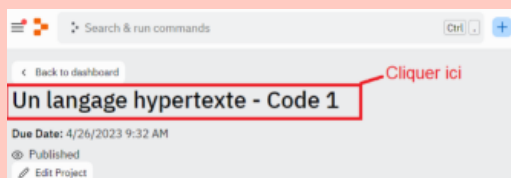
Pour avoir accès au code et à l'IDE intégré de cette leçon, vous devez :

- 1) Vous connecter à votre compte sur <https://replit.com/> (ou créer gratuitement votre compte)
- 2) Rejoindre la Team Code Studi du module via ce lien : <https://replit.com/teams/join/mmurvlgiplpxuasordloklllbqskoim-programmer-avec-javascript>

Une fois ces étapes effectuées, nous vous conseillons de rafraîchir votre navigateur si le code ne s'affiche pas.

En cas de problème, redémarrez votre navigateur et vérifiez que vous avez bien accepté les cookies de connexion nécessaires avant de recommencer la procédure.

Pour accéder au code dans votre cours, cliquez sur le nom du lien Replit dans la fenêtre. Par exemple :



II. Le format JSON

A. Le format JSON

Définition

JSON signifie JavaScript Object Notation. Il s'agit d'un format texte brut pour stocker et transporter des données légères entre ordinateurs. Il s'agit d'un langage indépendant, car dérivé de la syntaxe du JavaScript object notation. Le code pour lire et générer du JSON existe dans de nombreux langages de programmation.

Exemple Une chaîne (string) JSON

```
1 '{"name": "Charles", "age": 30, "car": null}'
```

[cf.]

C'est un exemple d'objet avec 3 propriétés :

- name
- age
- car

Chaque propriété a sa valeur. Si on parse (analyse) la chaîne JSON avec un programme JavaScript, on peut accéder aux données en tant qu'objet :

```
1 let jsonString = '{"name": "John", "age": 30}';
2 let obj = JSON.parse(jsonString);
3 let personName = obj.name;
4 let personAge = obj.age;
5 console.log(personName);
6 console.log(personAge);
```

[cf.]

Pourquoi utiliser JSON ? Le format JSON est syntaxiquement similaire au code de création d'objets JavaScript. Pour cette raison, un programme JavaScript peut facilement convertir des données JSON en objets JavaScript.

Depuis que le format est du texte brut uniquement, les données JSON peuvent facilement être envoyées entre ordinateurs et utilisées par n'importe quel langage. JavaScript a une fonction intégrée pour convertir les chaînes JSON en objets JavaScript, la fonction `JSON.parse()`. JavaScript a également une fonction pour convertir un objet en chaîne JSON, la fonction `JSON.stringify()`.

Stocker des données

Lors du stockage des données, les données doivent être dans un certain format, et, quel que soit l'endroit où vous choisissez de les stocker, le texte est toujours l'un des formats légaux. JSON rend possible de stocker des objets JavaScript sous forme de texte.

Attention Points importants à retenir

Nous pouvons recevoir du texte pur depuis un serveur et l'utiliser comme un objet JavaScript, envoyer un objet JavaScript à un serveur au format texte pur et travailler avec des données en tant qu'objets JavaScript, sans analyse ni traductions complexes.

La syntaxe JSON

La syntaxe JSON est un sous-ensemble de la syntaxe JavaScript. Voici les 4 règles de la syntaxe :

- Les données sont dans des paires nom/Valeur.
- Les données sont séparées par des virgules.
- Les accolades contiennent des objets.
- Les crochets contiennent des tableaux.

Une paire nom/valeur se compose d'un nom de champ (entre guillemets doubles), suivi de deux-points, suivi d'une valeur. Le format JSON est presque identique aux objets JavaScript. Les clés doivent être des chaînes, écrites avec des guillemets :

Une chaîne au format JSON :

```
1 {"name": "Charles"}
```

Un objet au langage JavaScript :

```
1 {name: "Charles"}
```

Le type de fichier pour les fichiers JSON est « .json ».

Les valeurs JSON

Dans JSON, les valeurs doivent être l'un des types de données suivantes :

- Une chaîne (string)
- Un nombre (number)
- Un objet (object)
- Un tableau (array)
- Un booléen (boolean - true/false)
- Null

Dans JavaScript, les valeurs peuvent être toutes les valeurs ci-dessus, plus toute autre expression JavaScript valide, y compris une fonction (function), une date (date) et undefined.

JSON vs XML

JSON et XML peuvent être utilisés pour recevoir des données d'un serveur web. Les deux présentent des spécificités non négligeables pour rédiger et transporter les données de la façon la plus claire qui soit.

Exemple

Les exemples JSON et XML suivants définissent tous deux un objet "employees", avec un tableau de 3 employés.

Objet avec tableau au format JSON :

```
1 {"employees": [  
2   { "firstName": "Charles", "lastName": "Dupont" },  
3   { "firstName": "Marie", "lastName": "Leclerc" },  
4   { "firstName": "Pierre", "lastName": "Lefranc" }  
5 ]}
```

[cf.]

Objet au format XML :

```
1 <employees>  
2   <employee>  
3     <firstName>Charles</firstName> <lastName>Dupont</lastName>  
4   </employee>  
5   <employee>  
6     <firstName>Marie</firstName> <lastName>Leclerc</lastName>  
7   </employee>  
8   <employee>  
9     <firstName>Pierre</firstName> <lastName>Lefranc</lastName>  
10  </employee>  
11 </employees>
```

[cf.]

Ces deux exemples mettent en avant que le JSON est comme le XML parce que « *autodescriptifs* », hiérarchiques (valeurs dans des valeurs), peuvent être analysés et utilisés par de nombreux langages de programmation. Ils peuvent également être récupérés avec un XMLHttpRequest.

Aussi, ces 2 exemples démontrent que le JSON est différent du XML parce que JSON n'utilise pas de balise de fin, est plus court, plus rapide à lire et écrire, et peut utiliser des tableaux.

Surtout, la plus grande différence est que XML doit être analysé (parse) avec un analyseur (parser) XML. JSON peut être analysé par une fonction JavaScript standard.

Attention Points importants à retenir

XML est plus difficile à analyser qu'un JSON. JSON est analysé dans un objet JavaScript prêt à l'emploi.

Les types de données JSON

Les chaînes (strings) dans JSON doivent être écrites avec des guillemets doubles. Les nombres (numbers) dans JSON doivent être un entier (integer) ou une virgule flottante (floating point). Les valeurs dans JSON peuvent être des objets (objects), des tableaux (arrays), des booléens (boolean) true/false, ou peuvent être null.

number JSON : { "age" : 30 }

object JSON :

```
{
  "employees": { "name": "Charles", "age": 30, "city": "Paris" }
}
```

array JSON :

```
{
  "employees": [ "Charles", "Marie", "Pierre" ]
}
```

boolean : { "rain": true }

null JSON : { "middlename": null }

Attention

Les objets en tant que valeurs dans JSON doivent suivre la syntaxe JSON.

JSON.parse()/JSON.stringify()/Stocker des données

Une utilisation courante de JSON consiste à échanger des données vers/depuis un serveur web. Lors de la réception de données d'un serveur web, les données sont toujours une chaîne. Analysez les données avec JSON.parse() et les données deviennent un objet JavaScript. Aussi, convertissez un objet JavaScript en chaîne avec JSON.stringify().

Lors du stockage des données, les données doivent être dans un certain format, et, quel que soit l'endroit où vous choisissez de les stocker, le texte est toujours l'un des formats légaux. JSON permet de stocker des objets JavaScript sous forme de texte.

Nous allons voir ces trois points dans cette vidéo.

Nous venons de voir comment analyser des données JSON afin qu'ils deviennent des objets JavaScript avec JSON.parse(). Aussi, nous venons de voir la conversion des objets JavaScript en chaîne JSON avec JSON.stringify(). Pour finir, le stockage et la récupération des données avec localStorage.

Attention Exceptions

Les objets date ne sont pas autorisés dans JSON. Si vous devez inclure une date, écrivez-la sous forme de chaîne. Vous pouvez le reconvertir en objet date ultérieurement.

Aussi, vous devez éviter d'utiliser les fonctions dans JSON, les fonctions perdront leur portée et vous devrez utiliser eval() pour les reconvertir en fonctions.

Serveur JSON

Vous pouvez faire une requête JSON au serveur en utilisant une requête Ajax. Tant que la réponse du serveur est écrite au format JSON, vous pouvez analyser la chaîne dans un objet JavaScript. Il faut utiliser XMLHttpRequest pour obtenir des données du serveur :

```

1 <!DOCTYPE html>
2 <html>
3   <body>
4     <p id="demo"></p>
5
6     <script>
7       const xmlhttp = new XMLHttpRequest();
8       xmlhttp.onload = function() {
9         const myObj = JSON.parse(this.responseText);
10        document.getElementById("demo").innerHTML = myObj.name;
11      }
12      xmlhttp.open("GET", "file.json");
13      xmlhttp.send();
14    </script>
15
16  </body>
17 </html>

```

Lors de l'utilisation de JSON.parse() sur JSON dérivé d'un tableau, la méthode renverra un tableau JavaScript au lieu d'un objet JavaScript.

```

1 <!DOCTYPE html>
2 <html>
3   <body>
4     <p id="demo"></p>
5
6     <script>
7       const xmlhttp = new XMLHttpRequest();
8       xmlhttp.onload = function() {
9         const myArr = JSON.parse(this.responseText);
10        document.getElementById("demo").innerHTML = JSON.stringify(myArr, null, 2);
11      }
12      xmlhttp.open("GET", "car.json", true);
13      xmlhttp.send();
14    </script>
15  </body>
16 </html>

```

[cf.]

B. Exercice : Quiz

[solution n°1 p.17]

Question 1

Que signifie JSON ?

- ☐ Jirascript object notation
- ☐ JavaScript Object Notation
- ☐ Java object notation

Question 2

Quelles sont les 4 règles de syntaxe JSON ? Plusieurs réponses sont possibles :

- ☐ Les données sont dans des paires nom/valeur
- ☐ Les données sont séparées par des virgules
- ☐ Les puces contiennent des numéros
- ☐ Les accolades contiennent des objets
- ☐ Les crochets contiennent des tableaux

Question 3

Quelle est la fonction intégrée à JavaScript pour convertir un objet en chaîne JSON ?

- ☐ string()
- ☐ stringify()
- ☐ JSON.stringify()

Question 4

Quelle est la fonction intégrée à JavaScript pour convertir une chaîne JSON en objet JavaScript ?

- ☐ parse()
- ☐ JSON.parse()
- ☐ JSON()

Question 5

```
1 const xmlhttp = new XMLHttpRequest();
2 xmlhttp.onload = function() {
3   const myObj = JSON.parse(this.responseText);
4   document.getElementById("demo").innerHTML =
5     myObj[0];
6 };
7 xmlhttp.open("GET", "car.json", true);
8 xmlhttp.send();
```

Lorsque je fais cette requête AJAX avec le JSON : ["Ford", "Renault", "Audi", "Fiat"]. Quelle sera la réponse ?

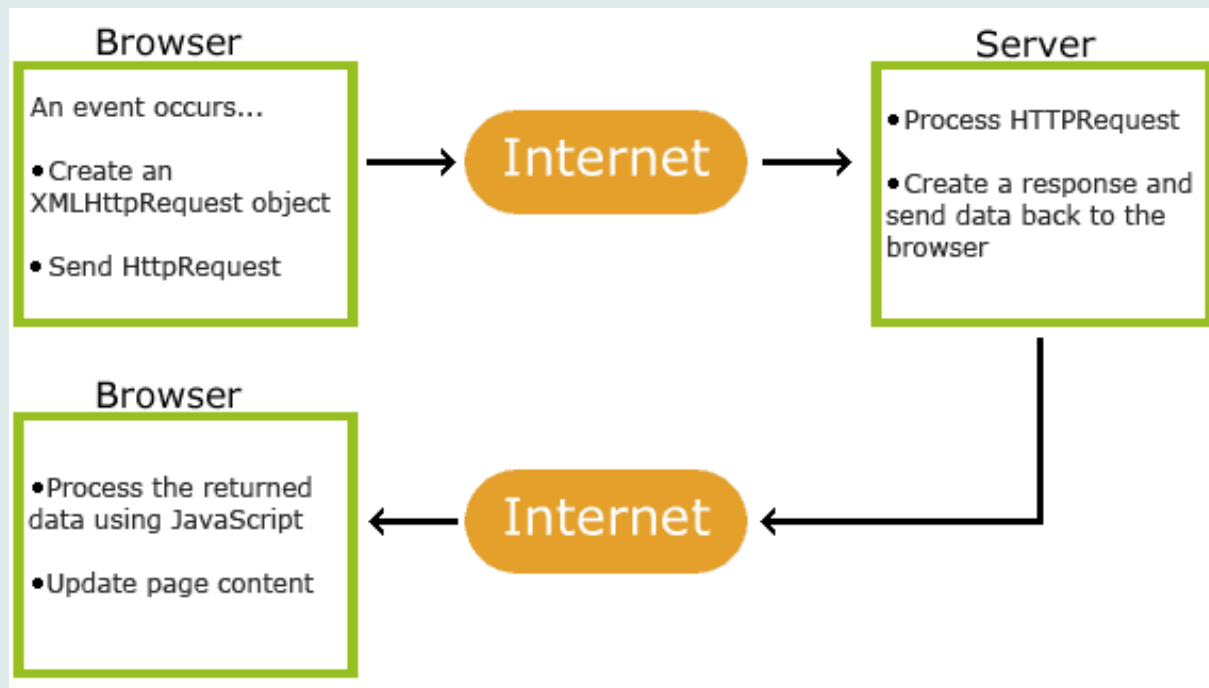
- ☐ "Ford"
- ☐ "Renault"
- ☐ "Audi"
- ☐ "Fiat"

III. AJAX

A. AJAX

Définition

AJAX signifie **A**synchronous **J**avascript **A**nd **X**ML. Il ne s'agit pas d'un langage de programmation. C'est une combinaison de requêtes (request) intégrées au navigateur et de JavaScript et HTML DOM pour afficher ou utiliser les données. L'acronyme peut prêter à confusion. Malgré son nom, AJAX peut transporter des données XML, mais également d'autres formats de texte comme JSON ou txt.



Source : w³schools¹

Comment fonctionne AJAX ?

Pour répondre à cette question, voici un schéma.

Un événement a lieu dans le navigateur, une requête HTTP est créée et envoyée au serveur. Le serveur traite la requête HTTP, génère une réponse et envoie les données en retour au navigateur. Les données envoyées par le serveur sont lues par JavaScript et la page est mise à jour.

XMLHttpRequest

La pierre angulaire d'AJAX est l'objet XMLHttpRequest. Tous les navigateurs modernes le prennent en charge. Il peut être utilisé pour échanger des données avec un serveur en arrière-plan. Cela signifie qu'il est possible de mettre à jour des parties d'une page web, sans recharger la page.

Aussi, il faut savoir que pour des raisons de sécurité les navigateurs n'autorisent pas l'accès entre les domaines. Cela signifie que la page web et le fichier XML, ou texte, qu'elle tente de charger doivent se trouver sur le même serveur. Si vous souhaitez charger les fichiers XML, ils doivent se trouver sur votre propre serveur.

¹ <https://www.w3schools.com/>

Également, l'objet XMLHttpRequest peut avoir différentes méthodes. Voici un tableau pour résumer ces méthodes :

Méthode	Description
new XMLHttpRequest()	Crée un nouvel objet XMLHttpRequest
abort()	Annule la demande en cours
getAllResponseHeaders()	Renvoie les informations d'en-tête
getResponseHeader()	Renvoie une information spécifique d'en-tête
open(method, url, async, user, psw)	Spécifie la demande method : la requête de type GET ou POST url : l'emplacement du fichier async : true(asynchrone) ou false(synchrone) user : nom d'utilisateur facultatif psw : mot de passe facultatif
send()	Envoie la requête au serveur Utilisé pour les requêtes GET
send(string)	Envoie la requête au serveur Utilisé pour les requêtes POST
setRequestHeader()	Ajoute une étiquette/valeur à l'en-tête à envoyer

L'objet XMLHttpRequest a également différentes propriétés. Voici un tableau pour les résumer :

Propriété	Description
onreadystatechange	Définit une fonction à appeler lorsque la propriété readyState change
readyState	Contient le statut de XMLHttpRequest 0 : requête non initialisée 1 : connexion au serveur établie 2 : requête reçue 3 : traitement en cours 4 : requête achevée et la réponse est prête
responseText	Renvoie les données sous forme de chaîne
responseXML	Renvoie les données sous forme de données XML

Propriété	Description
status	<p>Renvoie le numéro d'état d'une requête</p> <p>200 : « OK »</p> <p>403 : « Forbidden »</p> <p>404 : « Not Found »</p> <p>Pour une liste complète, cliquez sur ce lien : HTTP response status codes¹</p>
statusText	Renvoie le texte d'état (par exemple « OK » ou « Not Found »)

Envoyer une requête au serveur avec GET ou POST

Pour envoyer une requête à un serveur, nous utilisons les méthodes `open()` et `send()` de l'objet `XMLHttpRequest`, en utilisant soit GET ou POST. Il faut savoir que GET est simple et plus rapide que POST, et peut être utilisé dans la plupart des cas. Pour autant, il faut toujours utiliser POST lorsqu'un fichier en cache n'est pas une option (mettre à jour un fichier ou une base de données sur le serveur), envoyer une grande quantité de données au serveur (POST n'a pas de limitation de taille), et envoyer des entrées utilisateur (qui peuvent contenir des caractères inconnus). POST est plus robuste et sécurisé que GET.

Exemple Effectuer une requête au serveur avec POST

Dans cet exemple, à partir des éléments que nous avons vus au-dessus, vous devez effectuer une requête POST avec l'objet `XMLHttpRequest`. Dans ce cas, vous allez créer un fichier `.txt` et le "consommer". Les données du fichier `.txt` devront apparaître en cliquant sur un bouton. Aussi, il faudra que la requête soit asynchrone et ajouter la fonction `"onreadystatechange"`. Pour cela, il faudra vous référer au tableau sur les propriétés de l'objet `XMLHttpRequest`.

Explication :

1 - Nous créons un fichier texte.

```
1 <p>Salut, je m'appelle Charles, j'ai 34 ans et j'habite Paris.</p>
2 <p>Aussi, j'apprends AJAX.</p>
```

2 - Nous créons notre fichier HTML avec notre bouton.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width">
6   <title>Le format JSON et AJAX</title>
7   <link href="style.css" rel="stylesheet" type="text/css" />
8 </head>
9 <body>
10
11 <div id="demo">
12 <h1>L'objet XMLHttpRequest</h1>
13 <button type="button" onclick="loadDoc()">Voir le texte</button>
```

¹ <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

```
14 </div>
15
16 </body>
17 </html>
```

3 - Ensuite, nous créons notre script.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width">
6   <title>Le format JSON et AJAX</title>
7   <link href="style.css" rel="stylesheet" type="text/css" />
8 </head>
9 <body>
10
11 <div id="demo">
12   <h1>L'objet XMLHttpRequest</h1>
13   <button type="button" onclick="loadDoc()">Voir le texte</button>
14 </div>
15
16 <script>
17   //Création de la fonction
18   function loadDoc() {
19     var xhttp = new XMLHttpRequest();
20     xhttp.onreadystatechange = function() {
21       if (this.readyState == 4 && this.status == 200) {
22         document.getElementById("demo").innerHTML =
23           this.responseText;
24       }
25     };
26     xhttp.open("GET", "demo.txt", true);
27     xhttp.send();
28   }
29 </script>
30
31 </body>
32 </html>
```

[cf.]

4 - Voici le résultat dans le navigateur.

Salut, je m'appelle Charles, j'ai 34 ans et j'habite Paris.

Aussi, j'apprends AJAX.

Attention

Les requêtes XMLHttpRequest synchrones (async = false) ne sont pas recommandées parce que JavaScript va s'arrêter jusqu'à ce que la réponse du serveur soit prête. Si le serveur est occupé ou lent, l'application "freeze" s'arrête. C'est pour cette raison que cette requête est en train d'être supprimée de la norme web.

Attention

Les versions d'Internet Explorer antérieures à IE 10 ne prennent pas en charge l'objet XMLHttpRequest, qui est la base des requêtes AJAX.

Réponse du serveur

Dans cette vidéo, nous allons voir comment obtenir une réponse du serveur à travers différents exemples.

Le fichier XML

AJAX peut être utilisé de façon interactive avec les fichiers XML. C'est ce que nous allons voir dans cette vidéo.

Nous avons vu dans cette vidéo comment afficher les données d'un fichier .xml sous forme de tableau. Ainsi, les possibilités pour créer des applications interactives en utilisant XML, HTTP, DOM et JavaScript sont vastes.

B. Exercice : Quiz

[solution n°2 p.18]

Question 1

Que signifie AJAX ?

- ☐ Asynchronous Javascript And XML
- ☐ Synchronous Javascript And XML
- ☐ Async Javascript And XML

Question 2

Quels types de données AJAX peut-il transporter ? Plusieurs réponses sont possibles.

- ☐ .xml
- ☐ .json
- ☐ .xls
- ☐ .txt

Question 3

Quelle méthode est utilisée pour les requêtes GET envoyées au serveur ?

- ☐ abort()
- ☐ send()
- ☐ send(string)

Question 4

Quel est le type de requête qu'il faut toujours utiliser dans les méthodes open() et send() ?

- ☐ POST
- ☐ GET
- ☐ XMLHttpRequest

Question 5

Quelle est la fonction de la propriété `onreadystatechange` dans l'objet `XMLHttpRequest` ?

- ☐ Elle définit une fonction appelée chaque fois que l'état de la requête change
- ☐ Elle définit une fonction appelée lorsque la requête est en cours d'envoi
- ☐ Elle définit une fonction appelée lorsqu'une erreur survient lors de l'envoi de la requête
- ☐ Elle définit une fonction appelée lorsque la requête est terminée et que la réponse est prête à être traitée

IV. Essentiel

L'utilisation d'AJAX (Asynchronous JavaScript and XML) est une technique de programmation qui a révolutionné le développement des applications web modernes. Cette technique permet de mettre à jour dynamiquement les contenus d'une page web sans avoir besoin de la recharger complètement, offrant ainsi une expérience utilisateur plus réactive et plus fluide.

L'objet `XMLHttpRequest` en JavaScript, qui permet de créer des requêtes HTTP asynchrones depuis une page web vers un serveur web, a rendu possible l'utilisation d'AJAX. Cela signifie que la page web peut envoyer des requêtes au serveur sans que l'utilisateur ait à quitter la page ou à attendre une nouvelle page pour voir les résultats.

En combinant AJAX avec les technologies du web, telles que HTML, CSS et JavaScript, les développeurs peuvent créer des interfaces utilisateur riches et dynamiques, sans sacrifier la performance ou la qualité de l'expérience utilisateur.

L'utilisation d'AJAX est courante dans de nombreux domaines d'application, tels que les réseaux sociaux, les applications de messagerie, les outils de collaboration, les jeux en ligne, les applications de gestion de contenu et bien d'autres. Dans ces applications, AJAX est utilisé pour charger des contenus dynamiques, envoyer des messages, effectuer des recherches, soumettre des formulaires et mettre à jour les données de manière asynchrone.

V. Auto-évaluation

A. Exercice

Votre société de développement vous confie une "user story" (une fonctionnalité) pour l'un de ses clients. L'utilisateur doit pouvoir voir une liste de villes avec le code postal où se trouvent des dépôts en denrées alimentaires dans son département, sous forme de tableau. Ensuite, lorsqu'il clique sur une ville, cela affiche des informations sur l'adresse, date et horaire des dépôts ouverts au public. Il vous faudra consommer le fichier XML avec une requête AJAX.

Fichier XML donné :

```
1 <DEPARTMENT>
2   <CITY>
3     <NAME>Vernon</NAME>
4     <ZIP>27200</ZIP>
5     <ADDRESS>30 rue des carreaux</ADDRESS>
6     <DAY>03/05/23</DAY>
7     <HOUR>14H00 - 18h00</HOUR>
8   </CITY>
9   <CITY>
10    <NAME>Gaillon</NAME>
11    <ZIP>27600</ZIP>
12    <ADDRESS> 10 rue des joyeux</ADDRESS>
13    <DAY>13/05/23</DAY>
14    <HOUR>14H00 - 18h00</HOUR>
15  </CITY>
16  <CITY>
17    <NAME>Louviers</NAME>
18    <ZIP>27350</ZIP>
```

```

19      <ADRESS>60 avenue des vaches</ADRESS>
20      <DAY>08/05/23/</DAY>
21      <HOUR>14H00 - 18h00</HOUR>
22  </CITY>
23  <CITY>
24      <NAME>Gisors</NAME>
25      <ZIP>27420</ZIP>
26      <ADRESS>Rue Victor Leray</ADRESS>
27      <DAY>30/05/23/</DAY>
28      <HOUR>14H00 - 18h00</HOUR>
29  </CITY>
30  <CITY>
31      <NAME>Aubevoye</NAME>
32      <ZIP>27360</ZIP>
33      <ADRESS>Le Clos Maurice</ADRESS>
34      <DAY>23/05/23/</DAY>
35      <HOUR>14H00 - 18h00</HOUR>
36  </CITY>
37 </DEPARTMENT>

```

Fichier HTML donné :

```

1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <title>Exercice</title>
8   </head>
9   <style>
10     table,
11     th,
12     td {
13       border: 1px solid black;
14       border-collapse: collapse;
15     }
16     th,
17     td {
18       padding: 5px;
19     }
20   </style>
21   <body>
22     <h1>
23       Rechercher un dépôt dans votre ville
24     </h1>
25
26     <br /><br />
27     <p id="showCity"></p>
28     <table id="demo"></table>
29   </body>
30 </html>

```

Question 1

[solution n°3 p.19]

Créer la requête AJAX et ajouter une fonction pour afficher les villes avec leurs codes postaux sous forme de tableau.

Question 2

[solution n°4 p.20]

Créer une fonction pour afficher les informations pour chaque ville lorsqu'on clique dessus.

B. Test

Exercice 1 : Quiz

[solution n°5 p.20]

Question 1

AJAX ne peut fonctionner qu'avec des données au format XML.

- ☐ Vrai
- ☐ Faux

Question 2

Les requêtes AJAX ne peuvent pas être annulées une fois lancées.

- ☐ Vrai
- ☐ Faux

Question 3

Les requêtes AJAX sont toujours synchrones.

- ☐ Vrai
- ☐ Faux

Question 4

Les requêtes AJAX ne peuvent pas être utilisées pour envoyer des fichiers au serveur.

- ☐ Vrai
- ☐ Faux

Question 5


Les requêtes AJAX ne sont pas compatibles avec les anciennes versions d'Internet Explorer.

- ☐ Vrai
- ☐ Faux

Solutions des exercices


Exercice p. 7 Solution n°1**Question 1**

Que signifie JSON ?

- ☐ Jirascript object notation
- ☒ JavaScript Object Notation
- ☐ Java object notation
-  L'acronyme JSON signifie « *JavaScript Object Notation* ».


Question 2

Quelles sont les 4 règles de syntaxe JSON ? Plusieurs réponses sont possibles :

- ☒ Les données sont dans des paires nom/valeur
- ☒ Les données sont séparées par des virgules
- ☐ Les puces contiennent des numéros
- ☒ Les accolades contiennent des objets
- ☒ Les crochets contiennent des tableaux
-  La syntaxe JSON respecte les 4 règles suivantes : les données sont présentées par paires nom/valeur et doivent être séparées par des virgules ; les accolades doivent contenir des objets ; enfin, les crochets doivent contenir des tableaux.


Question 3

Quelle est la fonction intégrée à JavaScript pour convertir un objet en chaîne JSON ?

- ☐ string()
- ☐ stringify()
- ☒ JSON.stringify()
-  La fonction JSON.stringify() est intégrée à JavaScript et permet de convertir un objet en chaîne JSON.

Question 4

Quelle est la fonction intégrée à JavaScript pour convertir une chaîne JSON en objet JavaScript ?

- ☐ parse()
- ☒ JSON.parse()
- ☐ JSON()
-  La fonction JSON.parse() est intégrée à JavaScript et permet de convertir une chaîne JSON en objet JavaScript.

Question 5

```
1 const xmlhttp = new XMLHttpRequest();
2 xmlhttp.onload = function() {
3   const myObj = JSON.parse(this.responseText);
4   document.getElementById(« demo »).innerHTML =
5     myObj[0];
6 };
7 xmlhttp.open("GET", "car.json", true);
8 xmlhttp.send();
```


Lorsque je fais cette requête AJAX avec le JSON : ["Ford", "Renault", "Audi", "Fiat"]. Quelle sera la réponse ?

☒ "Ford"

☐ "Renault"

☐ "Audi"

☐ "Fiat"

 Le fichier JSON est un tableau. Lorsque nous parcourons le premier élément du tableau avec la fonction JSON.parse() dans notre const myObj, nous obtenons "Ford".

Exercice p. 13 Solution n°2


Question 1

Que signifie AJAX ?

☒ Asynchronous Javascript And XML

☐ Synchronous Javascript And XML

☐ Async Javascript And XML

 AJAX (Asynchronous JavaScript and XML) est une technique de programmation qui permet de faire des échanges de données entre un serveur web et une page web, sans avoir à recharger complètement la page.

Question 2


Quels types de données AJAX peut-il transporter ? Plusieurs réponses sont possibles.

☒ .xml

☒ .json

☐ .xls

☒ .txt

 Malgré son nom, AJAX peut transporter des données .xml, .json, .txt.


Question 3

Quelle méthode est utilisée pour les requêtes GET envoyées au serveur ?

☐ abort()

☒ send()

☐ send(string)

 La fonction send() est une méthode de l'objet XMLHttpRequest en JavaScript, utilisée pour envoyer une requête HTTP asynchrone au serveur web.


Question 4

Quel est le type de requête qu'il faut toujours utiliser dans les méthodes open() et send() ?

☒ POST

☐ GET

☐ XMLHttpRequest

 La requête POST est plus robuste et sécurisée que GET.

Question 5


Quelle est la fonction de la propriété onreadystatechange dans l'objet XMLHttpRequest ?

☒ Elle définit une fonction appelée chaque fois que l'état de la requête change

☐ Elle définit une fonction appelée lorsque la requête est en cours d'envoi

☐ Elle définit une fonction appelée lorsqu'une erreur survient lors de l'envoi de la requête

☐ Elle définit une fonction appelée lorsque la requête est terminée et que la réponse est prête à être traitée

 La propriété onreadystatechange permet de spécifier une fonction de rappel qui sera appelée chaque fois que l'état de la requête change. La fonction de rappel peut être utilisée pour mettre à jour dynamiquement les contenus de la page web en fonction de la réponse du serveur.

p. 15 Solution n°3

```
1 <script>
2     //Notre requête AJAX
3     const xhttp = new XMLHttpRequest();
4     let city;
5     xhttp.onload = function(){
6         const xmlDoc = xhttp.responseXML;
7         city = xmlDoc.getElementsByTagName("CITY");
8         loadCity();
9     };
10    xhttp.open("GET", "city.xml");
11    xhttp.send();
12
13 //Notre fonction pour analyser notre XML et afficher sous forme de tableau
14 function loadCity() {
15     let table = "<tr><th>Ville</th><th>Code postal</th></tr>";
16     for (let i = 0; i < city.length; i++) {
17         table += "<tr onclick='displayCity(" + i + ")'><td>";
18         table += city[i].getElementsByTagName("NAME")[0].childNodes[0]
19             .nodeValue;
20         table += "</td><td>";
```

```

21     table += city[i].getElementsByTagName("ZIP")[0].childNodes[0]
22         .nodeValue;
23     table += "</td></tr>";
24 }
25 document.getElementById("demo").innerHTML = table;
26 }
27
28 </script>

```

p. 15 Solution n°4

```

1 <script>
2 //Notre fonction pour afficher les informations pour chaque ville
3 function displayCity(i) {
4     document.getElementById("showCity").innerHTML =
5         "Dépôt : " +
6         city[i].getElementsByTagName("NAME")[0].childNodes[0].nodeValue +
7         "<br>Code postal : " +
8         city[i].getElementsByTagName("ZIP")[0].childNodes[0].nodeValue +
9         "<br>Adresse : " +
10        city[i].getElementsByTagName("ADRESS")[0].childNodes[0].nodeValue +
11        "<br>Jour : " +
12        city[i].getElementsByTagName("DAY")[0].childNodes[0].nodeValue +
13        "<br>Horaire : " +
14        city[i].getElementsByTagName("HOUR")[0].childNodes[0].nodeValue;
15    }
16 </script>

```

[cf.]


Exercice p. 16 Solution n°5

Question 1

AJAX ne peut fonctionner qu'avec des données au format XML.

☐ Vrai

☒ Faux


 Bien que l'acronyme AJAX contienne « XML », AJAX peut fonctionner avec différents formats de données, tels que JSON, HTML et texte brut. JSON est d'ailleurs devenu le format de données le plus couramment utilisé dans les applications AJAX en raison de sa simplicité et de sa légèreté.

Question 2

Les requêtes AJAX ne peuvent pas être annulées une fois lancées.

☐ Vrai

☒ Faux


 Il est possible d'annuler une requête AJAX en utilisant la méthode abort() sur l'objet XMLHttpRequest. Cela permet d'arrêter une requête en cours d'exécution, par exemple si l'utilisateur souhaite interrompre le processus.

Question 3

Les requêtes AJAX sont toujours synchrones.

☐ Vrai

☒ Faux


 L'une des principales caractéristiques des requêtes AJAX est leur capacité à être asynchrones. Les requêtes asynchrones permettent d'exécuter plusieurs tâches simultanément sans bloquer l'exécution du reste du code. Bien qu'il soit possible de rendre une requête AJAX synchrone, cela n'est généralement pas recommandé, car cela peut bloquer l'interface utilisateur.

Question 4

Les requêtes AJAX ne peuvent pas être utilisées pour envoyer des fichiers au serveur.

☐ Vrai

☒ Faux


 Les requêtes AJAX peuvent être utilisées pour envoyer des fichiers au serveur en utilisant l'objet FormData. FormData est une interface qui permet de construire facilement un ensemble de paires clé-valeur représentant les données du formulaire. Il peut être utilisé avec XMLHttpRequest pour envoyer des fichiers via AJAX, offrant ainsi une expérience utilisateur plus fluide et interactive lors du téléchargement de fichiers.

Question 5

Les requêtes AJAX ne sont pas compatibles avec les anciennes versions d'Internet Explorer.

☒ Vrai

☐ Faux

 Les versions d'Internet Explorer plus anciennes que IE 10 ne prennent pas en charge l'objet XMLHttpRequest, qui est la base des requêtes AJAX. Pour assurer la compatibilité avec ces anciennes versions d'IE, les développeurs devaient utiliser un objet ActiveXObject spécifique à IE. Toutefois, cette approche est de moins en moins courante, car les anciennes versions d'IE sont désormais obsolètes et la plupart des développeurs se concentrent sur les navigateurs modernes qui prennent en charge les fonctionnalités AJAX standard.