|  |  |
| --- | --- |
| SLAM5 | TP JSON |

**Pré-requis**

* Visual Studio 2019, installé

Connaissance de base des concepts et des commandes de Git, comme les dépôts, les duplications (fork), les clones, les branches, l’indexation et la désindexation des changements, et l’envoi (push) de commits

* Un compte GitHub
* Avoir des connaissances de bases du DOM

**Travail à réaliser**

Le travail à réaliser ci-dessous doit être référencé dans git et synchronisé sur github.

Chaque nouvel exercice doit faire l’objet de la création d’une branche et doit être rattaché au master lorsque l’exercice fonctionne et avant de passer au suivant.

**Quelques recherches :**

Que veut dire Json ?

**JavaScript Object Notation**

Qu’est-ce que Json ?

**JSON est un**[**format de données**](https://fr.wikipedia.org/wiki/Format_de_donn%C3%A9es)**textuelles dérivé de la notation des**[**objets**](https://fr.wikipedia.org/wiki/Objet_(informatique))**du langage**[**JavaScript**](https://fr.wikipedia.org/wiki/JavaScript)**. Il permet de représenter de l’information structurée**

**Exemple de code Json**

{

"personnage" :

[

{ "nom" : "Klad",

"classe" : "geurrier",

"age" : 28,

"armes" : [ "épée", "bouclier" ]

},

{ "nom" : "Valery",

"classe" : "magicienne",

"age" : 35,

"armes" : [ "bâton" ]

}

]

}

**Attention :**

* JSON requiert non seulement que le nom (la clé) soit entre guillemets, mais aussi que les guillemets soient doubles
* les guillemets simples sont possibles sur les objets JavaScript, pas dans JSON.

**Comment stocker du json**

JSON étant du texte, on peut le stocker où l'on veut. Dans une base de données, dans un fichier texte, un stockage client (cookies ou localStorage) ou via son propre format de fichier qui utilise l'extension .json (qui est en gros un fichier texte avec une extension .json).

Une fois le contenu JSON stocké, il reste donc à pouvoir le récupérer et le parser de manière appropriée. Selon les langages, il y a diverses façons de récupérer et parser JSON, mais puisque nous nous intéressons au développement front-end, je vais montrer comment le faire en JavaScript.

let myJSON = '{"nom" : "Klad","classe" : "guerrier","age" : 28,"armes" : [ "épée", "bouclier" ] }';

let myNewJSON = JSON.parse(myJSON);

// logs a JavaScript object, not a string

console.log(myNewJSON);

Dans l'exemple ci-dessus, la chaîne de caractères JSON est stockée dans une variable. Dans un scénario réel le JSON pourrait venir d'une source externe ou d'un fichier JSON séparé.

**Exercice 1**

Créez une page *personnage.html* et testez le code ci-dessus, qu’observez-vous ?

Nous pouvons maintenant utiliser ce qu'on appelle la notation avec point (dot notation) pour accéder aux différentes parties de la donnée JSON.

console.log(myNewJSON.nom);

console.log(myNewJSON.classe);

console.log(myNewJSON.age);

console.log(myNewJSON.armes);

**Exercice 2**

Ajoutez dans votre page *personnage.html*, l’affichage des données ci-dessus.

Utilisez le DOM pour afficher directement les informations dans la page HTML.

Faites-en sorte d’afficher chacune des armes indépendamment.

**Travailler avec un fichier json**

Le fichier json est mis à disposition sur un serveur http.

**XMLHttpRequest**

Les objets XMLHttpRequest (XHR) permettent d'interagir avec des serveurs. On peut récupérer des données à partir d'une URL sans avoir à rafraîchir complètement la page. Cela permet à une page web d'être mise à jour sans perturber les actions de l'utilisateur. XMLHttpRequest est beaucoup utilisé par l'approche AJAX

Malgré son nom, XMLHttpRequest peut être utilisé afin de récupérer tout type de données et pas uniquement du XML.

Sa méthodeopenpermet de configurer la requête HTTP avant son lancement. Elle prend en paramètres le type de requête HTTP (le plus souventGET, POST ouPUT), l'URL cible, ainsi qu'un booléen indiquant si la requête sera asynchrone ou non (voir plus loin).

Sa méthodesendenvoie la requête HTTP vers l'URL cible fournie àopen. Elle prend en paramètre l'éventuelle information envoyée au serveur (requêtesPOSTouPUT), ou bien nulldans le cas d'une requêteGET.

Sa propriétér esponseText contient sous forme textuelle la réponse renvoyée par le serveur à la requête HTTP.

//Le code suivant permet d’envoyer une requête de lecture d’un fichier Json

var requestURL = 'lien vers le fichier json';

var request = new XMLHttpRequest();

request.open('GET', requestURL);

request.responseType = 'json';

request.send();

//Stocker la réponse dans une varible

//cette variable contiendra désormais l'objet JavaScript basé sur le JSON

request.onload = function() {

var persoObj = request.response; // la requete dans un objet

var persoList = persoObj['personnage']; // récupération de la liste

for (var i = 0; i < persoList.length; i++) { // parcours de la liste

console.log(persoList[i].nom);

}

}

**Exercice 3**

Nouveau projet.

Sur le serveur wamp, sous www créez un répertoire *personnage*.

Dans le répertoire *personnage*, créez un page *perso.html*.

Créez, un répertoire Json.

Dans ce répertoire créez un fichier *perso.json* dans lequel vous placerez l’exemple de fichier json vu en début de document.

Dans la page perso html affichez grace au dom les personnages l’un à côté de l’autre.