JSON

Sarah BEN OTHMAN

ISEN

25 Février 2016

JSON (JavaScript Object Notation)

- JSON (jison ou jayson, jason en anglais) est reconnu par JavaScript, après que l'ECMA ait définit la fonction *eval* qui parse le format, dans le standard ECMAScript, en 1999.
- Il a été popularisé par le développement d'Ajax.
- Il permet de sérialiser une structure de données au format texte. JSON est largement utilisé pour stocker des données ou échanger des données notamment sur Internet mais aussi entre les couches IHM et applicative/service d'une application.
- On sait qu'il s'agit d'un format de fichier alternatif à XML, et ce format a ses adeptes.

- Mais qu'est-ce exactement que JSON, et quels sont ses avantages?

JSON (JavaScript Object Notation)

- JSON se base sur deux structures:
  - Une collection de couples nom/valeur. Divers langages la réifient par un *objet*, un enregistrement, une structure, un dictionnaire, une table de hachage, une liste typée ou un tableau associatif.
  - Une liste de valeurs ordonnées. La plupart des langages la réifient par un *tableau*, un vecteur, une liste ou une suite.

Ces structures de données sont universelles. Pratiquement tous les langages de programmation modernes les proposent sous une forme ou une autre. Il est raisonnable qu'un format de données interchangeable avec des langages de programmation se base aussi sur ces structures.

La syntaxe de JSON

JSON supporte plusieurs types de données :

- Numérique : nombre entier ou flottant
- Chaîne de caractères : ensemble de caractères Unicode (sauf une double quote et un antislash) entouré par des doubles guillemets
- Booléen : true ou false
- Tableau : un ensemble ordonné de valeurs entouré par des crochets. Chaque valeur est séparée par un caractère virgule. Les types des valeurs d'un tableau peuvent être différents
- Object : est composé de paires clé/valeur, chacune étant séparée par une virgule, entourées par des accolades. Une clé est obligatoirement une chaîne de caractères. Une valeur peut être : littérale (chaîne de caractères, nombre, true, false, null), un objet ou un tableau. Une clé est séparée de sa valeur par un caractère deux points
- La valeur null

La syntaxe de JSON

Dans une chaîne de caractères, le caractère d'échappement est le caractère antislash qui permet notamment de représenter dans la chaîne de caractères :

- \" : une double quote
- \\ : un antislash
- \/ : un slash
- \b : un backspace
- \f : un formfeed
- \n : une nouvelle ligne
- \r : un retour chariot
- \t : une tabulation
- \unnnn : le caractère Unicode dont le numéro est nnnn

Exemple :

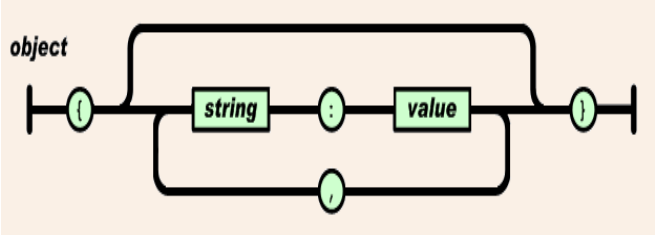
```
{ "prenom":"Jean-Michel" ,
"ville":"Paris" }
```

La syntaxe de JSON

- **L'objet**
  - Il contient un membre ou une liste de membres, chaque membre étant de la forme:

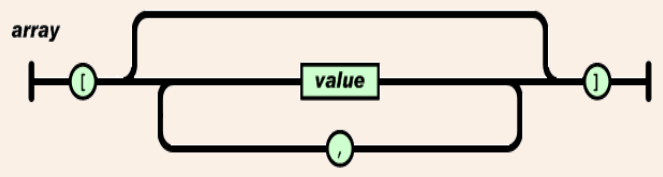
    "nom" : "valeur"
  - La syntaxe de l'objet est:

    { membre, membre, ... }

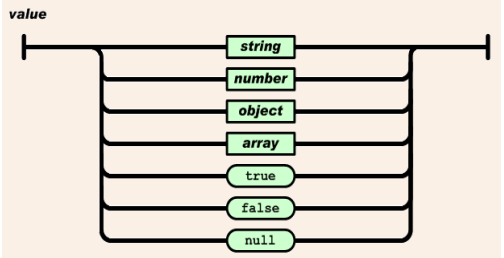


- **Le tableau**
  - Contient une ou plusieurs valeurs séparées par des virgules.

    [valeur, valeur, ...]



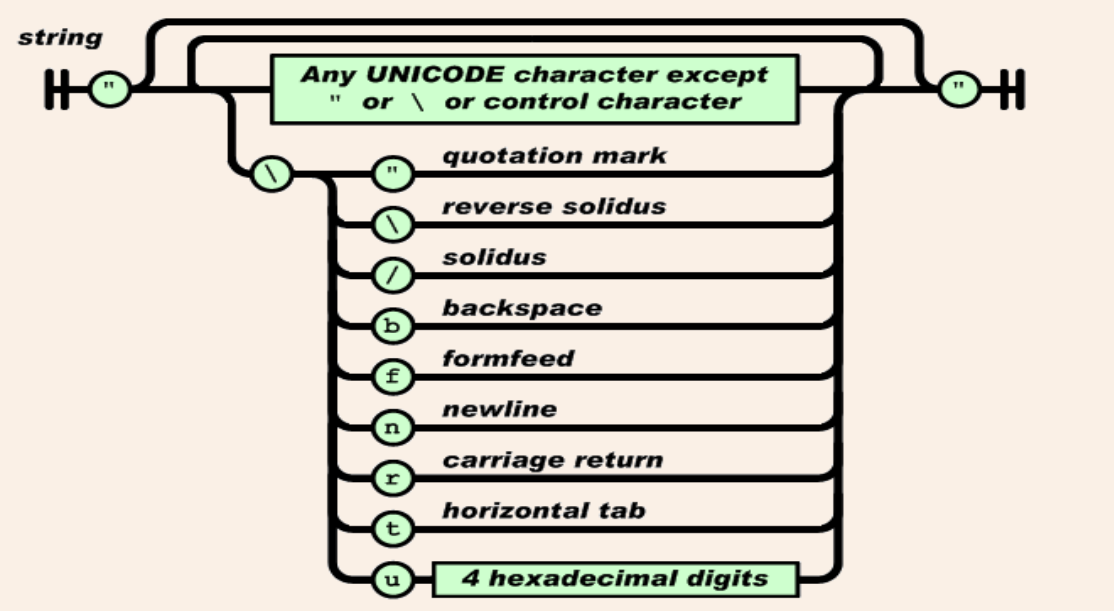
- **Les valeurs**
  - Une valeur peut être: un objet, un tableau, un littéral (chaîne, nombre, true, false, null).
  - Et il n'y a besoin de rien de plus à savoir pour créer un fichier JSON!



La syntaxe de JSON

- **Une chaîne de caractères:**

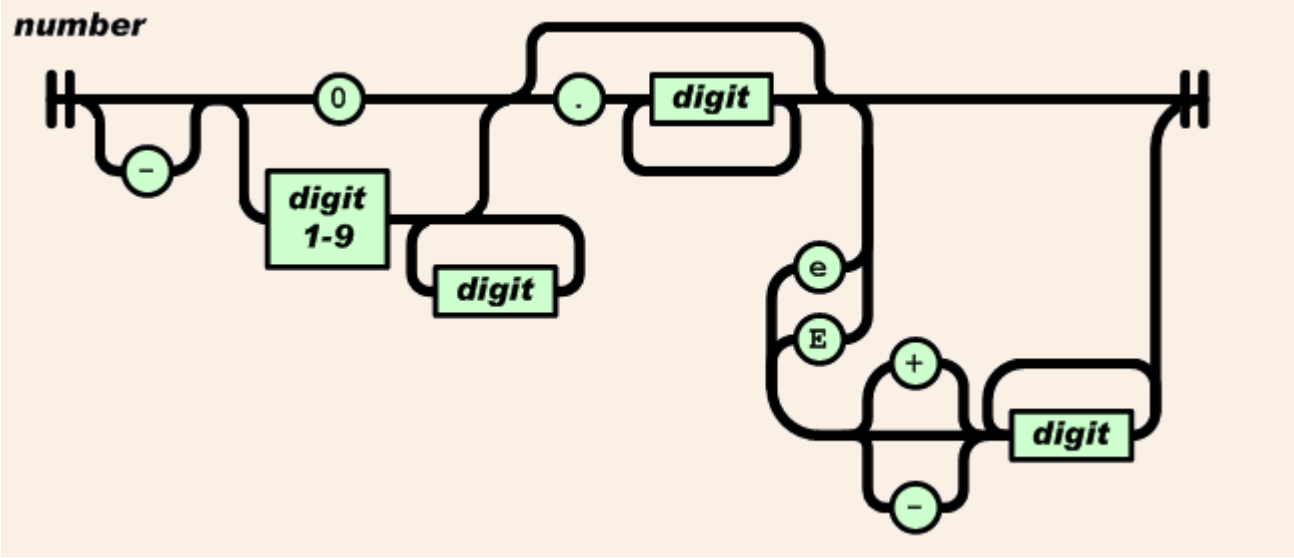
est une suite de zéro ou plus caractères Unicode, entre guillemets, et utilisant les échappements avec antislash. Un caractère est représenté par une chaîne d'un seul caractère. Une chaîne de caractères est très proche de ses équivalents en C ou en Java.



La syntaxe de JSON

- **Nombre**

Un *nombre* est très proche de ceux qu'on peut rencontrer en C ou en Java, sauf que les formats octal et hexadécimal ne sont pas utilisés.



JSON

- Le XML est une **calamité** à parser en JavaScript
- Le format JSON est beaucoup plus approprié
- JSON (JavaScript Object Notation) est un format de données générique qui utilise la notation des objets JavaScript pour transmettre de l'information structurée
- Exemple :

```
{"menu": {
    "id": "file",
    "value": "File",
    "popup": {
      "menuitem": [
            {"value": "New", "onclick": "CreateNewDoc()"},
            {"value": "Open", "onclick": "OpenDoc()"},
            {"value": "Close", "onclick": "CloseDoc()"}
    ] } }}
```

JSON (JavaScript Object Notation)

- permet de représenter de l'information structurée.
- Exemple :

```
{ "Image": {
 "Width": 800,
 "Height": 600,
 "Title": "Vue du 15ème étage",
 "Thumbnail": {
     "Url": "http://www.example.com/481989943",
     "Height": 125,
     "Width": "100" },
 "IDs": [116, 943, 234, 38793]
} }
```

Comment utiliser le format JSON

- **Coté client**
  JSON faisant partie de la norme JavaScript. Le contenu d'un fichier JSON, ou la définition de données dans ce format sont assignés à une variable, qui devient un objet du programme.
- **Coté serveur**
  Les fichiers au format JSON s'utilisent dans différents langages de programmation, notamment PHP et Java grâce à des parseurs qui permettent d'accéder au contenu, et éventuellement de le convertir en classes et attributs, dans ce langage.
- **L'échange de données**
  - inclusion directe du fichier dans la page HTML au même titre qu'un fichier .js de JavaScript.
  - Ou emploi de XMLHttpRequest.

  Le fichier JSON est parsé par la fonction JavaScript eval().

  Le transfert d'un fichier au serveur se fait par **XMLHttpRequest**. Le fichier au format texte est traité par le parseur du langage de programmation utilisant le fichier.

Exemple d'utilisation de JSON

- Le code XMLHttpRequest:

```
var req = new XMLHttpRequest();

req.open("GET", "fichier.json", true);

req.onreadystatechange = monCode;

req.send(null);
```

- Le code JavaScript:

```
function monCode()

{ if (req.readyState == 4)

     { var doc = eval('(' + req.responseText + ')'); } }
```

- Utilisation des données:

```
var nomMenu = document.getElementById('jsmenu');
     // trouver un champ

nomMenu.value = doc.menu.value;

// assigner une valeur au champ
```

- Accéder aux données :

```
doc.commands[0].title // lire la valeur de "title" dans le tableau

doc.commands[0].action // lire la valeur de "action" dans le tableau
```

Les avantages de JSON

- Type de données générique et abstrait pouvant
  - être représenté dans n'importe quel langage de programmation
  - représenter n'importe quelle donnée concrète.
- simple à mettre œuvre tout en étant complet.
- peu verbeux, lisible aussi bien par un humain que par une machine
- facile à apprendre, syntaxe réduite
- types de données sont connus et simples à décrire
- indépendant du langage de programmation (bien qu'utilisant une notation JavaScript)
- Le type MIME application/json est utilisé pour le transmettre par le protocole HTTP (notamment en Ajax)

Les avantages de JSON

- Vis-à-vis de JavaScript, un document JSON représente un objet, d'où son nom. Il est donc plus facile à interpréter qu'un XML.

```
var donnees = eval('('+donnees_json+')');
```

- Le site json.org fournit une liste de parseurs pour d'autres langages
- Il peut aussi être utilisé pour :
  - la sérialisation et déserialisation d'objets ;
  - l’encodage de documents ;

Les avantages de JSON

En utilisant un objet *JSON* il est possible de passer:

- de **Object à JSON** string en utilisant ***JSON.stringify()***
- de **JSON string à Object** en utilisant ***JSON.parse()***

```
var reality = '{"level": 1, "team": ["Ariadne", "Arthur", "Cobb", "Eames", "Fischer", "Saito", "Yusuf"], "dream": {"dreamer": "Yusuf"}}';
var realityObject = JSON.parse(reality); // JSON String to JS Object console.log(realityObject.team[0]); // => "Ariadne"
realityObject.level = 42;
realityObject.dream.dreamer = "Charles";
reality = JSON.stringify(realityObject); // JS Object to JSON String console.log(reality);
// Console output :
// {"level":42,"team":["Ariadne","Arthur","Cobb","Eames","Fischer","Saito","Yusuf"],"dream":{"dreamer":"Charles"}}
```

JSON vs. XML

Les avantages de JSON:

- La vitesse de traitement.
- La simplicité de mise en oeuvre.
  On n'a pas besoin de parser un fichier XML pour extraire des informations à travers le net, car JSON est reconnu nativement par JavaScript.
- Les contenus binaires peuvent être intégrés et échangés sur le net avec une représentation textuelle spéciale avec une commande comme: new Buffer(file).toString('base64').

Les avantages de XML:

- XML est extensible quand au langage, on peut créer des formats comme RSS, SVG.
- Il est largement utilisé et reconnu par tous les langages de programmation.
- Il est plus facile à lire et convient mieux pour les fichiers destinés aux non programmeurs.

Noter que XML aussi bien que JSON ne conviennent pas pour stocker directement des données binaires de taille importante.