Le format JSON

JSON, pour JavaScript Object Notation, est un format texte léger principalement conçu à des fins d'échange de données. JSON est un format qui permet de stocker des informations structurées. À la base, il s'agit d'une manière de représenter des données, similaire dans son objectif à XML ou YAML, mais souvent considérée comme plus lisible et succincte.

Qu'est-ce que JSON ?

Imaginez avoir besoin de décrire un livre. En anglais, on pourrait dire qu'il a un titre, un auteur et un certain nombre de pages. JSON prend cette capacité descriptive, voici à quoi cela pourrait ressembler :

```
{
      "title": "The Great Gatsby",
      "author": "F. Scott Fitzgerald",
      "pages": 180
}
```

Cette structure permet aux développeurs, applications et systèmes de comprendre et de transmettre des données de manière standardisée.

Bref Historique : Origine et Évolution

Pour apprécier pleinement JSON, il faut remonter à la fin des années 1990. Douglas Crockford, qui travaillait alors chez State Software, a d'abord spécifié et popularisé le format. Bien qu'il ait été introduit dans le cadre du langage JavaScript, sa simplicité et son adaptabilité ont conduit à sa large adoption dans des paysages de programmation variés.

Le nom « JavaScript Object Notation » pourrait laisser penser que c'est uniquement pour JavaScript, mais c'est une idée fausse courante. En raison de son élégance et de sa simplicité, JSON a trouvé sa place dans une myriade de langages de programmation comme Python, Ruby et Java, entre autres. Au fil du temps, JSON a été peaufiné, affiné et a dépassé sa portée initiale, mais reste largement fidèle à sa conception originale.

Comprendre La Syntaxe JSON

JSON, en son cœur, consiste à représenter des données structurées d'une manière à la fois lisible par l'homme et conviviale par la machine. Une compréhension approfondie de sa syntaxe vous permettra de représenter, transmettre et comprendre un vaste éventail d'informations. Décodeons les éléments constitutifs essentiels de la syntaxe JSON :

- Il ne doit exister qu'un seul élément père par document contenant tous les autres : un élément racine.

- Tout fichier JSON bien formé doit être :
  - soit un objet commençant par **{** et se terminant par **}**,
  - soit un tableau commençant par **[** et terminant par **]**.
- Cependant ils peuvent être vides, ainsi **[]** et **{}** sont des JSON valides.
- Les séparateurs utilisés entre deux paires/valeurs sont des virgules.
- Un objet JSON peut contenir d'autres objets JSON.
- Il ne peut pas y avoir d'éléments croisés.

Types de données : Nombres, Chaînes, Booléens, Tableaux, Objets, Nuls

1. CHIFFRES

En JSON, les nombres sont des chiffres qui peuvent être à la fois des nombres entiers et des nombres à virgule flottante. Ils n'ont pas besoin de guillemets autour d'eux.

Exemple :

```
{
      "age": 30,
      "weight": 68.5
}
```

2. CHAÎNES

Les chaînes sont des séquences de caractères entourées de guillemets doubles. Ils peuvent contenir des lettres, des chiffres et d'autres caractères.

Exemple:

```
{
	"nane": "John Doe",
	"profession": "Ingénieur"
}
```

3. BOOLÉEN

Ce type de données représente des valeurs de vérité et peut être `true` ou `false`.

Exemple:

```
{
	"isStudent": false,
	"hasLicence": true
}
```

4. TABLEAU

Les tableaux sont des collections ordonnées et peuvent contenir plusieurs valeurs, même de types de données différents.

Exemple:

```
{
	"colors": ["red", "green", "blue"],
	"score": [85, 90, 78.5]
}
```

5. OBJET

Les objets sont des collections de paires clé-valeur. Chaque clé est une chaîne, suivie de deux points, puis de sa valeur correspondante.

Exemple:

```
{
	"person": {
		"firstName": "Jane",
		"lastName": "Doe"
	}
}
```

6. NULLE

`Null` représente une valeur vide ou inexistante.

Exemple:

```
{
	"middleName": null
}
```

1. PAIRES CLÉ-VALEUR ET OBJETS

Un objet en JSON est une collection non ordonnée de paires clé-valeur. Les clés, représentées sous forme de chaînes, sont des identifiants uniques pour les valeurs qu'elles contiennent. Cette relation constitue la structure fondamentale de JSON.

Exemple:

```
{
	"firstName": "Jane",
	"lastName": "Doe",
	"age": 32,
	"isActor": true
```

```
}
```

Remarquez comment chaque clé est associée à sa valeur à l'aide de deux-points, et chaque paire clé-valeur est séparée par une virgule.

2. TABLEAUX ET STRUCTURES DE DONNÉES IMBRIQUÉES

Les tableaux jouent un rôle crucial dans la représentation de listes ordonnées de valeurs. Ils peuvent également être imbriqués dans des objets et contenir différents types de données.

Exemple:

```
{
      "books": [
            {
                  "title": "Moby Dick",
                  "author": "Herman Melville"
            },
            {
                  "title": "1984",
                  "author": "George Orwell"
            }
      ]
}
```

Dans l'exemple ci-dessus, « books » est un tableau contenant deux objets. Chaque objet représente un livre avec un titre et un auteur. Cette représentation illustre la puissance des structures imbriquées de JSON, permettant une représentation de données complexe.

En maîtrisant ces éléments fondamentaux de la syntaxe JSON, vous êtes sur la bonne voie pour devenir compétent dans la création, le décodage et la compréhension des structures de données basées sur JSON. Alors que JSON continue de dominer le domaine de l'échange de données, ces connaissances s'avéreront sans aucun doute inestimables.

Les principaux usages du JSON

1. CHARGEMENT ASYNCHRONES

Avec la montée en flèche des chargements asynchrones tels que l'AJAX (Asynchronous JavaScript And XML) dans le web actuel (qui ne provoquent pas le rechargement total de la page), il est devenu de plus en plus important de pouvoir charger des données organisées, de manière rapide et efficace.

Avec XML, le format JSON s'est montré adapté à ce type de besoins.

2. LES APIS

Des sociétés telles que Twitter, Facebook ou LinkedIn, offrent essentiellement des services basés sur l'échange d'informations, et font preuve d'un intérêt grandissant envers les moyens possibles pour distribuer ces données à des tiers.

Alors qu'il n'y a pas de domination totale d'un des deux formats (JSON ou XML) dans le domaine des APIs, on constate toutefois que JSON est en train de prendre le pas là où le format XML avait été pionnier.

3. LES BASES DE DONNÉES

Le JSON est très utilisé dans le domaine des bases de données NoSQL (MongoDB, CouchDB, Riak...).

On notera également qu'il est possible de soumettre des requêtes à des SGBDR tel que MySQL ou PostgreSQL et de récupérer une réponse en JSON.