UNIVERSITE VIRTUELLE DU SENEGAL

*foo nekk foofu la*

# Introduction à JSON

## Le format de données JSON



**El hadji Mamadou NGUER Enseignant chercheur en Informatique à l'UVS**

2020

# Chapitre 1 : Introduction à JSON

**Objectifs spécifiques :** A la suite de ce chapitre, l'étudiant doit être capable de:

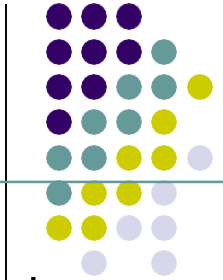
1. Présenter JSON,
2. Donner un exemple de données au format JSON
3. Enumérer les raisons d'utiliser JSON
4. Décrire à quoi peut servir JSON.
5. Comparer JSON et XML

# Chapitre1 : Introduction à JSON

**Plan du chapitre 1**:

1. Qu'est-ce que JSON?
2. Exemple de d'objet JSON
3. Pourquoi utiliser JSON ?
4. A quoi peut servir JSON ?
5. JSON vs XML

# Qu'est-ce que JSON?



- JSON est l'acronyme de **J**ava **S**cript **O**Bject **N**otation (Notation Objet issue de JavaScript)
- C'est un format léger d'échange de données
- Il est indépendant du langage *
- Et il est «auto-décrivant» et facile à comprendre
- Le format JSON a été initialement spécifié par Douglas Crockford** .

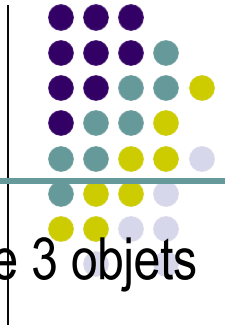
* La syntaxe JSON est dérivée de la syntaxe de notation d'objet de JavaScript, mais le format JSON est uniquement en texte. Le code de lecture et de génération de données JSON peut être écrit dans n'importe quel langage de programmation.

** Cest un programmeur et entrepreneur américain, notamment connu pour sa forte implication dans le développement du langage JavaScript et pour la création du format JSON



elhadjimamadou.nguer@uvs.edu.sn

# Exemple de d'objet JSON

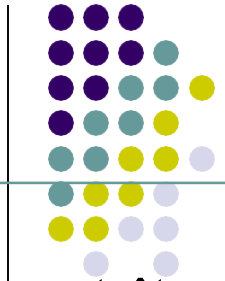


- L'exemple JSON suivant définit un objet employé contenant un tableau de 3 objets d'employés :

```
{
"employes":[
    {"prenom":"Saliou", "nom":"Ndiaye"},
    {"prenom":"Anna", "nom":"Sarr"},
    {"prenom":"Paul", "nom":"Sene"}
]
}
```

- Le format JSON est syntaxiquement identique au code pour créer des objets JavaScript.
- En raison de cette similitude, un programme JavaScript peut facilement convertir les données JSON en objets JavaScript natifs.

# Pourquoi utiliser JSON ?



- Étant donné que le format JSON est uniquement du texte, il peut facilement être envoyé vers et depuis un serveur et utilisé comme format de données par n'importe quel langage de programmation.

- JavaScript a une fonction intégrée pour convertir une chaîne, écrite au format JSON, en objets JavaScript natifs:

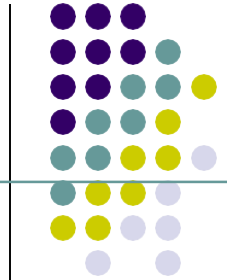
  JSON.parse()

- Ainsi, si vous recevez des données d'un serveur, au format JSON, vous pouvez les utiliser comme n'importe quel autre objet JavaScript.



elhadjimamadou.nguer@uvs.edu.sn

# A quoi peut servir JSON ?

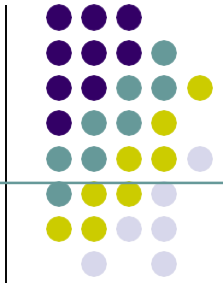


- **A échanger des données :**
  - Lors de l'échange de données entre un navigateur et un serveur, les données ne peuvent être que du texte.
  - JSON est du texte, et il est possible de convertir un objet JavaScript en JSON et vice versa lors d'échange de données avec le serveur.
  - De cette façon, on peut travailler avec les données en tant qu'objets JavaScript, sans analyse ni traduction compliquée.
- **A envoyer des données :**
  - Si vous avez des données stockées dans un objet JavaScript, vous pouvez convertir l'objet en JSON et l'envoyer à un serveur:

```
var myObj = {nom: "Juma", age: 31, ville: "Gosaas"};
var myJSON = JSON.stringify(myObj);
window.location = "demo_json.php?x=" + myJSON;
```

  - La fonction JSON.stringify() sera étudiée dans la suite.

# A quoi peut servir JSON ?



- **A recevoir des données :**
  - Si on reçoit des données au format JSON, on peut les convertir en objet JavaScript:

```
var myJSON = '{"nom":"Fatou", "age":31, "ville":"Thies"}';
var myObj = JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.name;
```

  - La fonction JSON.parse() sera étudiée dans la suite.

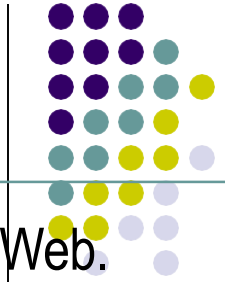
- **A stocker des données :**
  - Lors du stockage de données, les données doivent être dans un certain format, et quel que soit l'endroit où vous choisissez de les stocker, le texte est toujours l'un des formats légaux.

```
// Stockage des données:
myObj = {nom: "Cheikh", age: 31, city: "Touba"};
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

// Récupération des données :
text = localStorage.getItem("testJSON");
obj = JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

  - JSON permet de stocker des objets JavaScript sous forme de texte.

# JSON vs XML



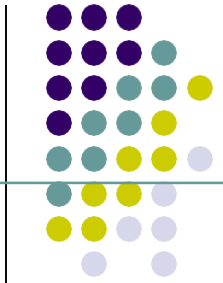
- JSON et XML peuvent être utilisés pour recevoir des données d'un serveur Web.
- Les exemples JSON et XML suivants définissent tous deux un objet Employés, avec un tableau de 3 employés:
- **Exemple :** (JSON)

```
{"employes":[
  { "prenom":"Patrick", "nom":"Lumbumba" },
  { "prenom":"Anna", "nom":"Sarr" },
  { "prenom":"Pierre", "nom":"Ngom" }
]}
```

- **Exemple :** (XML)

```
<employes>
  <employe>
    <prenom>Patrick</prenom> <nom>Lumbumba</nom>
  </employe>
  <employe>
    <prenom>Anna</prenom> <nom>Sarr</nom>
  </employe>
  <employe>
    <prenom>Pierre</prenom> <nom>Ngom</nom>
  </employe>
</employes>
```

# JSON vs XML



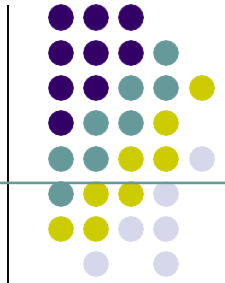
## Points commun entre JSON et XML :

- Ils sont "auto-descriptifs" (lisibles par l'homme)
- Ils sont hiérarchiques (valeurs dans les valeurs)
- Ils peuvent être analysés et utilisés par de nombreux langages de programmation
- Ils peuvent être récupérés avec un XMLHttpRequest

## Ce qui diffère JSON de XML :

- JSON n'utilise pas de balise de fin
- JSON est plus court
- JSON est plus rapide à lire et à écrire
- JSON peut utiliser des tableaux
- **La plus grande différence est que XML doit être analysé avec un analyseur XML, alors que JSON peut être analysé par une fonction JavaScript standard.**

# JSON vs XML



**Pourquoi JSON est meilleur que XML:**

- XML est beaucoup plus difficile à analyser que JSON.
- JSON est analysé dans un objet JavaScript prêt à l'emploi.
- Pour les applications AJAX, JSON est plus rapide et plus facile que XML:

- **Pour utiliser un document XML, on doit :**
  - Récupérer le document XML
  - Utiliser le DOM XML pour parcourir le document
  - Extraire des valeurs et stocker dans des variables

- **Pour utiliser un document JSON, on doit :**
  - Récupérer la chaîne JSON
  - Parser la chaîne JSON avec JSON.Parse