UNIVERSITE VIRTUELLE DU SENEGAL

foo nekk foofu la

Syntaxe et types de données JSON

Le format de données JSON

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



2019

Chapitre 2 : Syntaxe et types de données JSON

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

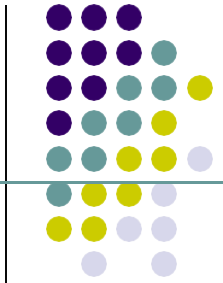
1. Décrire la syntaxe de JSON
2. Décrire les types de données dans JSON
3. Décrire les objets dans JSON
4. Décrire les tableaux dans JSON

Chapitre 2 : Syntaxe et types de données JSON

Plan de la séquence :

1. La syntaxe de JSON
2. Les types de données de JSON
3. Les objets dans JSON
4. Les tableaux dans JSON

La syntaxe de JSON



Règles de syntaxe de JSON

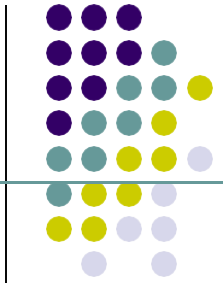
La syntaxe JSON est dérivée de la syntaxe de notation d'objet JavaScript ,:

- Les données sont dans les paires nom propriété / valeur
- Les données sont séparées par des virgules
- Les accolades entourent les objets
- Les crochets contiennent des tableaux

Données JSON

- Les données JSON sont écrites sous forme de paires nom / valeur.
- Une paire nom / valeur se compose d'un nom de champ (entre guillemets), suivi de deux points, suivi d'une valeur:
- Exemple : `"nom":"Jaara"`
- Les noms JSON nécessitent des guillemets doubles contrairement à JavaScript.

La syntaxe de JSON



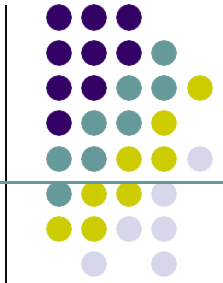
JSON évalué en objet JavaScript

- Le format JSON est presque identique aux objets JavaScript.
- En JSON, les clés doivent être des chaînes écrites entre guillemet : `{"nom":"Sey"}`
- En JavaScript, les clés peuvent être des chaînes, des nombres ou des noms d'identifiants : `{ nom:"Jaata" }`

Les valeurs JSON

- Dans JSON , les valeurs doivent être l'un des types de données suivants: **string**, **number**, **object** (JSON object), **array**, **boolean** et **null**
- En JavaScript, les valeurs peuvent être des types ci-dessus, ainsi que toute autre expression JavaScript valide, notamment : **function**, **date** et **undefined.**

La syntaxe de JSON



JSON utilise la syntaxe JavaScript

- Étant donné que la syntaxe JSON est dérivée de la notation d'objet JavaScript, très peu de logiciels supplémentaires sont nécessaires pour travailler avec JSON dans JavaScript.
- Avec JavaScript, on peut créer un objet et lui affecter des données, comme ceci:

```
var personne = { nom: "Sokhna", age: 31, ville: "Dakar" };
```

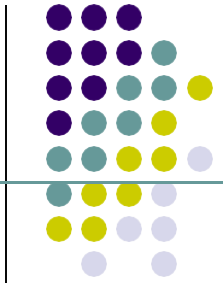
- Vous pouvez accéder à un objet JavaScript comme celui-ci :

  personne.nom; ou personne["nom"];

- Les données peuvent être modifiées comme ceci:

  personne.nom = "Gilaaye"; ou personne["nom"] = "Gilaaye";

La syntaxe de JSON



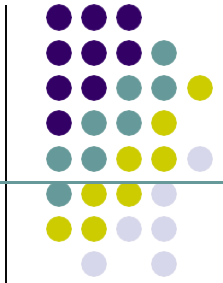
Tableaux JavaScript en JSON

- De la même manière que les objets JavaScript peuvent être utilisés comme JSON, les tableaux JavaScript peuvent également être utilisés comme JSON.
- On verra les tableaux au format JSON dans la suite.

Fichier JSON

- Le type de fichier pour les fichiers JSON est ".json"
- Le type MIME pour le texte JSON est "application / json"

Les types de données



Types de données valides

- Dans JSON, les valeurs doivent être de l'un des types de données suivants:
  - String
  - number
  - object (JSON object)
  - array
  - boolean
  - null

Les valeurs de type **function**, **date** et **undefined** ne sont permises dans JSON

- **Exemples :**

- Valeur de type String : `{ "nom":"Jaara" }`

- Valeur de type Number : Elle peut être un entier ou un réel : `{"note":10}`

- Valeur de type Object :

```
{
"employe":{ "nom":"Jaara", "age":30, "ville":"Dakar" }
}
```

Les valeurs de type objet doivent suivre les mêmes règles que les objets JSON.

Les types de données

Types de données valides

- **Exemples :**

  - Valeur de type Array :

```
{
  "employes":[ "Jaara", "Anna", "Paate" ]
}
```

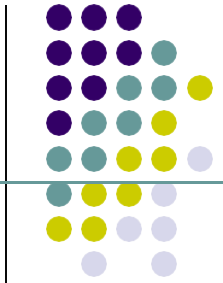
  - Valeur de type Boolean : Elle peut avoir les valeurs true ou false :

```
{"choix":true}
```

  - Valeur de type Null :

```
{ "nom":null }
```

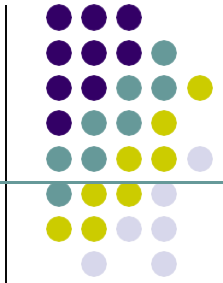
Les objets dans JSON



La syntaxe :

- Les objets JSON sont entourés d'accolades {}.
- Ils sont écrits sous forme de paires propriété / valeur.
- Chaque propriété est chaîne et chaque valeur doit être un type de données JSON valide (chaîne, nombre, objet, tableau, booléen ou null).
- Une propriété et sa valeur sont séparées par deux points.
- Les paires propriété / valeur sont séparées par une virgule.
- **Exemple :** `{ "nom":"Sokhna", "age":30, "voiture":null }`

Les objets dans JSON



Accès aux valeurs d'un objet :

- On peut accéder aux valeurs des objets en utilisant la notation avec point (.) ou la notation avec crochet ([]):

Exemple :

```
myObj = { "nom":"Sokhna", "age":30, "voiture":null };
x = myObj.nom;  //ou
x = myObj["nom"];
```

Boucler sur un objet :

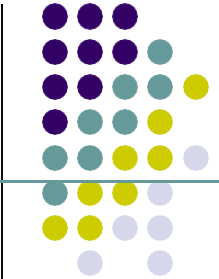
- Vous pouvez parcourir les propriétés des objets en utilisant la boucle **for-in**:

```
myObj = { "nom":"Sokhna", "age":30, "voiture":null };
for (x in myObj) {
  document.getElementById("demo").innerHTML += x;
}
```

- La notation avec crochets peut aussi être utilisée pour accéder aux valeurs :

```
for (x in myObj) {
  document.getElementById("demo").innerHTML += myObj[x];
}
```

Les objets dans JSON



Imbrication de objet

- Un objet JSON peut avoir comme valeur un autre objet JSON.

- On peut accéder aux objets JSON imbriqués à l'aide de la notation par points ou de celle par crochet :

```
myObj = {
  "nom":"Sokhna",
  "age":30,
  "voitures": {
    "voiture1":"Ford",
    "voiture2":"BMW",
    "voiture3":"Fiat"
  }
 }
```

```
x = myObj.voitures.voiture2;
// or:
x = myObj.voitures["voiture2"];
```

Modification d'objet

- On peut utiliser la notation par points ou par crochets pour modifier n'importe quelle valeur dans un objet JSON:

```
myObj.voitures.voiture2 = "Mercedes";
```

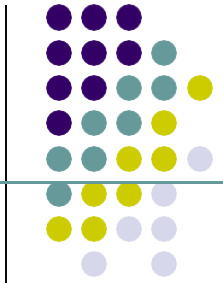
```
myObj.voitures["voiture2"] = "Mercedes";
```

Suppression de propriété d'objet

- Le mot clé delete est utilisé pour supprimer les propriétés d'un objet JSON:

```
delete myObj.voitures.voiture2;
```

Les tableaux dans JSON



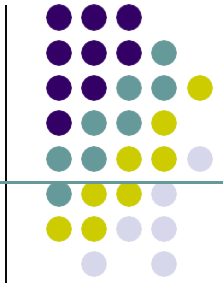
- Les tableaux en JSON sont presque les mêmes qu'en JavaScript.
- Cependant, les valeurs de tableau doivent être de type chaîne, nombre, objet, tableau, booléen ou null .
- Alors qu'en JavaScript, elles peuvent être, en plus des valeurs ci-dessus, toute autre expression JavaScript valide, y compris les fonctions, les dates et même undefined.
- **Exemple :**

```
[ "Ford", "BMW", "Fiat",30 ]
```

- Un tableau peut être la valeur d'une propriété d'un objet JSON :

```
{
"nom":"Sokhna",
"age":30,
"voitures":[ "Ford", "BMW", "Fiat" ]
}
```

Les tableaux dans JSON



Accès aux valeurs d'un tableau JSON

- On accéde aux valeurs d'un tableau en utilisant les numéros d'index:

```
x = myObj.voitures[0];
```

- On peut les accèder aussi en utilisant une boucle **for** :

```
for (i = 0; i < myObj.voitures.length; i++) {
  x += myObj.voitures[i];
}
```

- On peut également utiliser une boucle **for-in** :

```
for (i in myObj.voitures) {
  x += myObj.voitures[i];
}
```

Les tableaux dans JSON

Accès aux valeurs d'un tableau JSON

- Un tableau peut avoir comme valeur un autre tableau ou même un autre objet JSON:

```
myObj = {
  "nom":"Sokhna",
  "age":30,
  "voitures": [
    { "nom":"Ford", "modeles":[ "Fiesta", "Focus", "Mustang" ] },
    { "nom":"BMW", "modeles":[ "320", "X3", "X5" ] },
    { "nom":"Fiat", "modeles":[ "500", "Panda" ] }
  ]
 }
```

- Dans ce cas une boucle **for-in** est utilisée pour chaque tableau:

```
for (i in myObj.voitures) {
  x += "<h1>" + myObj.voitures[i].nom + "</h1>";
  for (j in myObj.voitures[i].modeles) {
    x += myObj.voitures[i].modeles[j];
  }
}
```

Les tableaux dans JSON

Mise à jour des valeurs d'un tableau JSON

- Pour modifier une valeur d'un tableau, on utilise le numéro d'index :

```
myObj.voitures[1] = "Mercedes";
```

- Et pour supprimer une valeur, le mot-clé delete est utilisé:

```
delete myObj.voitures[1];
```