ISEN | école d'ingénieurs
ALL IS DIGITAL!

# Communications Web

## *JSON*

*ISEN Ouest*

*thibault.napoleon@isen-ouest.yncrea.fr*

# Sommaire I

ISEN OUEST

# Sommaire I

ISEN OUEST

# JSON

ISEN OUEST

## Pourquoi ?

**Comment échanger des informations structurées entre deux entités ne partageant pas le même langage ?**

⇒ En les décrivant par une chaîne de caractères où l'information est structurée par des étiquettes qui permettent d'interpréter les divers éléments.

**Comment ?**

Créer un langage de description au format texte comprenant :

- Une syntaxe simple, peu verbeuse et figée.
- La prise en charge des types de bases.
- La possibilité d'avoir des types structurés (tableaux et dictionnaires).

# JSON

## Définition

ISEN OUEST

### Définition

*JSON* pour *JavaScript Object Notation*, est un format de données textuelles basé sur la notation des objets en *JavaScript**. Il permet d'échanger simplement des données entre deux entités (client-serveur, application-fichier...) grâce à une structuration de l'information.

**Normalisation :**

*JSON* a été créé par Douglas Crockford dans les années 2000 (entre 2002 et 2005).

Désormais cette notation est décrite par la *RFC* (c.-à-d.*Request for comments*) 8259 ainsi que par le standard *ECMA-404*.

* Malgré son nom, c'est une notation indépendante de tout langage de programmation.

# Sommaire I

ISEN OUEST

# Le format

ISEN OUEST

## Typage

**Quatre types de bases :**

Les types de bases du format *JSON* sont les suivants :

- **Chaîne de caractères** : séquence de $0$ ou plus caractères.
- **Nombre** : nombre décimal ou flottant signé.
- **Booléen** : *true* ou *false*.
- ***null*** : valeur vide.

**Deux types d'éléments structurés :**

Un document *JSON* ne comprend que deux types d'éléments structurels :

- **Tableau** : liste ordonnée de valeurs.
- **Dictionnaire** : ensemble de paires nom/valeur (aussi appelé *objet*)

# Le format

ISEN OUEST

Syntaxe des types structurés

**Les tableaux :**

Ils contiennent un ensemble de valeurs compris entre **crochets** et séparé par des **virgules** :

```
[valeur1, valeur2, ...]
```

Chaque valeur du tableau est un élément de type *JSON* (dictionnaire, tableau, chaîne de caractères, nombre, booléen, *null*).

**Les dictionnaires :**

Ils contiennent un ensemble de clés/valeurs compris entre **accolades** et séparé par des **virgules**. Le séparateur du couple est **deux points** :

```
{"cle1": valeur1, "cle2": valeur2, ...}
```

Les clés sont des chaînes de caractères et les valeurs des éléments de type *JSON*.

# Le format

ISEN OUEST

## Exemple

```json
{
  "Produits": [{
    "Nom": "Verre",
    "Prix": 9.99,
    "Quantité": 23,
    "En stock": true,
    "Couleurs": ["Vert", "Rouge", "Bleu"],
    "Marque": "Ikea"
  }, {
    "Nom": "Assiette",
    "Prix": 12.99,
    "Quantité": 0,
    "En stock": false,
    "Couleurs": ["Blanc"],
    "Marque": null
  }],
  "Magasin": "www.vaisselle.fr"
}
```

# Sommaire I

ISEN OUEST

# Syntaxe

*PHP*

ISEN OUEST

## Encodage (*json_encode*) :

```php
$data = array(1, true, "isen", array("fruit" => "kiwi",
  "nb" => 2));
$json = json_encode($data);
print_r($json); // [1, true, "isen", {"fruit": "kiwi",
                // "nb": 2}]
```

## Décodage (*json_decode*) :

```php
$json = '[1, true, "isen", {"fruit": "kiwi", "nb": 2}]';
$data = json_decode($json);
print_r($data); // Array([0] => 1 [1] => 1 [2] => isen
                // [3] => Array([fruit => kiwi [nb] => 2]))
```

# Syntaxe

*JavaScript*

ISEN OUEST

## Encodage (*JSON.stringify*) :

```
var data = [1, true, 'isen', {'fruit': 'kiwi', 'nb': 2}];
var json = JSON.stringify(data);
console.log(json); // [1, true, "isen", {"fruit: "kiwi",
                   // "nb": 2}]
```

## Décodage (*JSON.parse*) :

```
var json = '[1, true, "isen", {"fruit": "kiwi", "nb": 2}]';
var data = JSON.parse(json);
console.log(data); // [0:1, 1:true, 2:"isen",
                   // 3:{fruit:"kiwi", nombre:2}]
```

# Sommaire I

ISEN OUEST

# Conclusion

ISEN OUEST

## Remarques

**Les commentaires :**

Étant donné que la notation *JSON* est un format d'échange axé donné, les commentaires, qui sont des métadonnées, **ne sont pas admis**.

Cependant, certains interpréteurs, non stricts, autorisent l'ajout de commentaires avec : /, // ou /*.

**Les commentaires sont donc à éviter ! !**

**La sécurité :**

Les requêtes *AJAX* utilisent souvent la notation *JSON* pour échanger des informations entre le client et le serveur et vice-versa. De ce fait, le *JSON* est souvent un vecteur de vol de données lors de l'exploitation d'une faille de type *Cross-Site Request Forgery*.

# Conclusion

ISEN OUEST

Avantages et inconvénients de cette notation

**Avantages :**

- Réduit fortement la bande passante nécessaire.
- Peu verbeuse donc très léger et rapide à traiter.
- Les types de données sont connus de façon exhaustive.
- Facile à apprendre (syntaxe réduite et non extensible).

**Inconvénients :**

- Type de données très restreint.
- Syntaxe non extensible.
- Peu sécurisé (pas de taille pour les chaînes...).
- Pas de *metadata* (commentaires, auteur, *timestamp*...).

ISEN | école d'ingénieurs
ALL IS DIGITAL!

# Avez vous des questions ?