INF5190 - MVC - Modèle

Jean-Philippe Caissy

18 septembre 2019

Sérialisation de données

Comment s'assurer la représentation des données échangés?

- Utilisation de formats standardisés
- Indépendant du langage de programmation et du système d'exploitation

Sérialisation de données

JSON

JSON : Javascript Object Notation

- Simple et léger
- Structure de donnée avancé (listes, dictionnaires)
- Format lisible par un humain
- Très populaire et popularisé par Javascript

Sérialisation de données

JSON

Syntaxe

- Système de clé-valeur
- Types de données supportées :
  - Entier
  - Flottant
  - Chaine de caractère
  - Booléan
  - Liste (array)
  - Objet (dictionnaire)
  - null

Sérialisation de données

JSON

Syntaxe

Types de bases

```
{
    "spouse" : null,
    "name" : "John Smith",
    "isAlive" : true,
    "age" : 27,
    "hourly_salary": 18.5
}
```

Sérialisation de données

JSON

Syntaxe

Liste

- Délimité par des crochets `[ ]` et séparés par une virgule.
- Une liste est un élément **ordonnée**. Peut être vide.
- Peut contenir n'importe quel autre type

```
{
   "name" : "John Smith",
   "children": [],
   "phoneNumbers" : ["212 555-1234", "123 456-7890"]
}
```

Sérialisation de données

JSON

Objets

- Délimité par des accolades { }
- Les valeurs ne sont pas ordonnées.

```
{
   "name" : "John Smith",
   "address" : {
      "postalCode" : "10021-3100",
      "streetAddress" : "21 2nd Street",
      "state" : "NY",
      "city" : "New York"
   }
}
```

Sérialisation de données

JSON

Objets

Les objets peuvent être imbriqués et contenir n'importe quel type de valeurs

```
{
   "address" : {
      "postalCode" : "10021-3100",
      "streetAddress" : "21 2nd Street",
      "state" : "NY",
      "city" : "New York",
      "phone": {
          "type": "mobile",
          "number": "212 555-1234"
      }
   }
}
```

Sérialisation de données

JSON

Validation

- Un objet JSON avec un erreur de syntaxe est invalide.
- https://jsonformatter.org/

Sérialisation de données

XML

XML : Extensible Markup Langage

- Popularisé dans les annéess 1990
- Très utilisés par de vieux systèmes et protocoles
- Plus lourd que XML, mais encore plus flexible
- Permet une validation basée sur un schéma existant

Sérialisation de données

XML

- Arbre d'éléments
  - Un seul élément racine
  - Notion d'éléments parents et enfants
- Aucune notion de types de données

Sérialisation de données

XML

Syntaxe

```
<note>
    <to>
        <name>John Smith</name>
        <email>john.smith@example.com</emaail>
    </to>
    <from>Toto</from>
    <subject>Rappel important!</subject>
    <body>
        Ne pas oublier l'évènement de la
        semaine prochaine
    </body>
</note>
```

Sérialisation de données

XML

Attributs

```
<note>
    <to email="john.smith@example.com">John Smith</to>
    </to>
    <from>Toto</from>
    <subject priority="1">Rappel important!</subject>
    <body>
        Ne pas oublier l'évènement de la
        semaine prochaine
    </body>
</note>
```

HTML est un sous-ensemble de XML!

Bases de données

- Base de données relationnelles
  - Basé sur un modèle *relationel* des données représentés par un schéma
  - L'interaction se fait souvent avec SQL
  - Exemple : Oracle Database, Microsoft SQL Server, MySQL, PostgreSQL
- Base de données non-relationnelles (NoSQL)
  - Aucun schéma relationnel
  - Interface simple de stockage et récupération de donnée
  - Existe sous plusieurs formes :
    - Stockage de documents (MongoDB, CouchDB)
    - Clé-valeur (key-value) (Redis, Memcache, ZooKeeper)
    - Base de donnée de graphe (Neo4J)

Base de données

ACID

- Atomicité : une transaction se fait au complet ou pas du tout
- Cohérence : les transactions respectent les contraintes en place et mènera un état valide à un nouvel état valide
- Isolation : chaque transaction est exécuté comme si elle était unique et linéaire; aucune dépendance aux autres transactions
- Durable : une transaction confirmée va demeurer enregistré, même en cas de panne

Les base de données relationnelles supportent toutes les paramètres ACID. Ce n'est pas toujours le cas des base de données non-relationnelles.

Bases de données

Stockage de documents

- Objectif : stocker, récupérer et gérer de l'information sous forme de documents
- Un document représente des données semi-structurés
  - Ex: XML, JSON
- Différence majeure entre une base de donnée relationnelle :
  - Dans une BD relationnelle les informations peuvent être stockés dans des tables différentes et les objets représentés à travers plusieurs tablse
  - Dans une base de donnée de documents, l'information est représenté au complet dans un seul objet.
    - Chaque objet peut être différent des autres objets
    - Élimine le besoin relationnel

Bases de données

Clé-valeur (key-value)

- Objectif : stocker et récupérer des données associées à une clé
- Rapide et performant
  - Aucune opération complexe
  - Complexité algorithmique pour récupérer les éléments de `O(1)`
- Certaines base de données gardent les données seulement en mémoire (RAM)
- Souvent utilisé par des systèmes de caching : la récupération des données est très rapide et efficace

Bases de données

Graphes

- Base de donnée représentant les informations sous forme de graphe
- Opérations possibles sur les noeuds et arêtes
- Structure relationnelle : les noeuds sont joints entre-eux
  - Chaque relation contient aucune, ou plusieurs propriétés