TP N°. 4 : DOM XML

Manipulation des Documents XML en JavaScript

M. L. FOUGHALI

Université 20 Août 1955, Skikda
3ème Année Licence 3 Informatique, Option ISIL
Données Semi-Structurées (DSS)

abwaelouey@gmail.com

Lundi 22 Avril 2024

Plan de l'exposé

DOM XML (Browser) - Hello World

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>DOM XML</title>
</head>
<body>
  <p id="result"></p>

  <script>
    const xmlText = `<?xml version="1.0" encoding="UTF-8"?> <message>Hello World
        </message>`;

    var parser = new DOMParser();
    var xmlDoc = parser.parseFromString(xmlText, "text/xml");

    document.getElementById("result").innerHTML = getHelloWorld(xmlDoc);

    function getHelloWorld(xmlDoc) {
      var message = xmlDoc.getElementsByTagName("message")[0];
      return message.textContent;
    }
  </script>
</body>
</html>
```

Installation de Node.js et JSDOM sous Windows

Les étapes à suivre pour installer JSDOM, sous node.js :

1. Visitez : `https://nodejs.org/`
2. Téléchargez (en ce jour) : `node-v20.12.2-x64.msi`
3. Lancez l'exécutable et suivez les instructions.
4. Confirmez l'installation :

   ```
   D:\> node --version
   ```
5. Installez JSDOM :

   ```
   D:\> npm install jsdom
   ```

JSDOM - Hello World

```
1  const { JSDOM } = require("jsdom");
2
3  const xmlText = '<?xml version="1.0" encoding="UTF-8"?><
       message>Hello World</message>';
4
5  const dom = new JSDOM(xmlText, { contentType: "text/xml" });
6  const xmlDoc = dom.window.document;
7
8  function getHelloWorld(xmlDoc) {
9    var message = xmlDoc.getElementsByTagName("message")[0];
10   return message.textContent;
11 }
12
13 console.log(getHelloWorld(xmlDoc));
```

Explication :

Ligne 1 : Import de JSDOM avec déstructuration.

Document XML Test

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="web">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
  </book>

     . . .       <!-- Ce fichier est incomplet !  -->

  <book category="web" cover="paperback">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

Exemple 1 : Le nombre de livres

```
function countBooks(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    return books.length;
}
```

Explication :

Cette fonction utilise la méthode `getElementsByTagName` pour récupérer tous les éléments <book> et renvoie le nombre total de ces éléments, donnant ainsi le nombre total de livres.

Exemple 2 : Le nombre d'auteurs

```
function countAuthors(xmlDoc) {
    var authors = xmlDoc.getElementsByTagName("author");
    return authors.length;
}
```

Explication :

Utilisant `getElementsByTagName`, cette fonction compte tous les éléments `<author>` présents dans le document, reflétant le nombre total d'auteurs.

Exemple 3 : Nombre de livres en français

```
function countFrenchBooks(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("title");
    var count = 0;
    for (var i = 0; i < books.length; i++) {
        if (books[i].getAttribute("lang") === "fr") {
            count++;
        }
    }
    return count;
}
```

Explication :

Cette fonction parcourt tous les éléments <book> et utilise `getAttribute` pour vérifier la langue de chaque livre. Elle compte ceux dont l'attribut `lang` est égal à `"fr"`.

Exemple 4 : La liste des titres

```
function listTitles(xmlDoc) {
    var titles = xmlDoc.getElementsByTagName("title");
    var list = [];
    for (var i = 0; i < titles.length; i++) {
        list.push(titles[i].textContent);
    }
    return list;
}
```

Explication :

En récupérant tous les éléments <title>, cette fonction assemble une liste des titres de livres, démontrant comment accéder et manipuler le contenu textuel des nœuds.

Exemple 5 : La liste des auteurs

```
function listAuthors(xmlDoc) {
    var authors = xmlDoc.getElementsByTagName("author");
    var list = [];
    for (var i = 0; i < authors.length; i++) {
        list.push(authors[i].textContent);
    }
    return list;
}
```

Explication :

Cette fonction collecte les noms de tous les auteurs à partir des éléments <author>, illustrant l'extraction de données textuelles à partir de nœuds.

Exemple 6 : Les livres dont le prix est supérieur à 30.00

```
function expensiveBooks(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    var list = [];
    for (var i = 0; i < books.length; i++) {
        var price = parseFloat(books[i].getElementsByTagName
            ("price")[0].textContent);
        if (price > 30.00) {
            list.push(books[i].getElementsByTagName("title")
                [0].textContent);
        }
    }
    return list;
}
```

Explication :

En examinant chaque élément <book>, cette fonction vérifie le <price> de chaque livre et ajoute le titre à la liste si le prix dépasse 30.00.

Exemple 7 : Les livres par année de publication

```
function listBooksByYear(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    var booksByYear = {};
    for (var i = 0; i < books.length; i++) {
        var year = books[i].getElementsByTagName("year")[0].
            textContent;
        if (!booksByYear[year]) {
            booksByYear[year] = [];
        }
        booksByYear[year].push(books[i].getElementsByTagName
            ("title")[0].textContent);
    }
    console.log(booksByYear);
    return booksByYear;
}
```

Explication :

Cette fonction organise les livres par leur année de publication. Elle crée un objet où chaque clé est une année et chaque valeur est une liste de titres publiés cette année.

Exemple 8 : Le nombre de livres par catégorie

```
function countBooksByCategory(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    var countByCategory = {};
    for (var i = 0; i < books.length; i++) {
        var category = books[i].getAttribute("category");
        if (!countByCategory[category]) {
            countByCategory[category] = 0;
        }
        countByCategory[category]++;
    }
    console.log(countByCategory);
    return countByCategory;
}
```

Explication :

Cette fonction compte le nombre de livres dans chaque catégorie en utilisant l'attribut category de chaque élément <book>, offrant une vue d'ensemble de la répartition des genres ou sujets dans la bibliothèque XML.

Exemple 9 : Les livres ayant plusieurs auteurs

```
function listBooksWithMultipleAuthors(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    var list = [];
    for (var i = 0; i < books.length; i++) {
        var authors = books[i].getElementsByTagName("author"
            );
        if (authors.length > 1) {
            list.push(books[i].getElementsByTagName("title")
                [0].textContent);
        }
    }
    return list;
}
```

Explication :

Cette fonction identifie les livres qui ont plusieurs auteurs en vérifiant le nombre d'éléments <author> pour chaque livre. Elle fournit une liste des titres de ces livres, mettant en évidence la collaboration entre auteurs dans certains ouvrages.

Exemple 10 : Le prix moyen des livres

```
function calculateAveragePrice(xmlDoc) {
    var books = xmlDoc.getElementsByTagName("book");
    var total = 0;
    for (var i = 0; i < books.length; i++) {
        var price = parseFloat(books[i].getElementsByTagName
            ("price")[0].textContent);
        total += price;
    }
    var average = total / books.length;
    return average.toFixed(2);
}
```

Explication :

Cette fonction calcule le prix moyen des livres en additionnant tous les prix et en divisant par le nombre total de livres, permettant une analyse coût-bénéfice des acquisitions de la bibliothèque.

URLs des sources

Merci pour votre attention !

Des questions ?

Les codes sources :

La source DOM XML (Browser)

Une fois la page ouverte, enregistrez-la !

La source JSDOM (node.js)