1. Introduction

Le but du TP est d'apprendre à modéliser différentes informations en XML. Pour cela, le TP est en deux parties. Dans un premier temps, vous allez observer des fichiers XML existants, puis vous allez créer vos propres fichiers.

Vous avez plusieurs éditeurs à votre disposition, du plus simple au plus complexe :

- `gedit` est l'éditeur minimal pour travailler avec XML.
- `geany` peut compléter les balises et faire une coloration syntaxique qui aide bien à écrire des documents corrects. Son plugin `PrettyPrinter XML` (menu `Outils`) est très pratique pour bien aligner et indenter les balises.
- `XML Copy Editor` est un éditeur complet qui intègre de nombreux outils (validation) qui nous seront utiles au prochain TP. Vous pouvez modifier les préférences, par exemple changer la police de caractère, par exemple `DejaVu Sans Mono` qui est plus lisible.

Commencez par créer un dossier `tp1` pour ce TP et travaillez dedans.

2. Étude de fichiers XML

De nombreuses applications emploient la norme XML pour enregistrer les documents. On va étudier rapidement quelques formats parmi les plus simples pour se faire une idée de ce qu'on peut représenter.

2.1. Format DocBook

Le format DocBook permet d'écrire des livres sans se soucier de la mise en page. C'est un peu similaire à LaTeX, tout en offrant de plus grandes capacités d'extraction des informations du document. En effet, les balises qui sont employées sont sémantiques : elles indiquent la signification, le rôle de ce qui est écrit. En LaTeX, les balises indiquent seulement la mise en page. Pour travailler sur un document DocBook, il est recommandé d'utiliser un éditeur spécialisé comme XXE. Open/Libre Office Writer peut enregistrer au format DocBook version 4.

Télécharger le fichier `livre.xml` et `livre.pdf` qui est sa traduction en pdf. Vous pourriez vous-même effectuer la traduction en pdf en faisant `docbook2pdf livre.xml`, mais cette commande n'est pas installée. Pour les fichiers XML, soit vous les ouvrez dans un nouvel onglet, puis affichez le source avec `CTRL-U`, soit vous les téléchargez avec `wget`, ou un clic droit `enregistrez la cible du lien sous...`.

Ouvrir le fichier livre.xml avec `XML Copy Editor` ou `geany`. Ce livre est ultra-simple. Les balises sont très compréhensibles. Ce qui nous intéresse, c'est la structure XML du document source. Quand vous voyez un document XML, vous devez vous poser les questions suivantes :

- Quelle est la racine du document ?
- Quel est le namespace du document ?
- Quels sont les éléments qu'on trouve sous la racine ?

Sur ce document, voici d'autres questions à se poser :

- Quels sont les attributs d'un élément `chapter` ?

- Aurait-il été possible que le titre d'un chapitre soit un attribut de l'élément `<chapter>` au lieu d'un élément `<title>` enfant ?
- Imaginons maintenant que le document contienne aussi les titres en anglais des mêmes chapitres. Comment pourrait-on les faire coexister avec les titres en français ?
- Inversement, serait-il possible que le numéro de révision d'un chapitre soit un sous-élément plutôt qu'un attribut ? Et justement, si on voulait rajouter un commentaire au numéro de révision, comment pourrait-on faire ?

2.2. Format SVG

Ce format représente des images vectorielles, c'est à dire des dessins créés avec des figures géométriques : lignes, polygones, splines, etc. remplies ou non. Les images vectorielles sont à opposer aux images matricielles composées de pixels. Une image vectorielle peut-être agrandie indéfiniment sans montrer de défauts, par contre elle ne peut pas représenter une photographie efficacement (parce que les appareils photos sont matriciels, aucun n'est vectoriel, ça reste à inventer).

Le format SVG permet de très nombreuses choses. On va seulement en avoir un aperçu. Télécharger le fichier `dessin.svg`. Pour l'afficher, il suffit simplement de l'ouvrir dans le navigateur ou avec le visionneur d'images (clic droit, ouvrir avec...). Vous pouvez zoomer dessus autant que vous voulez, les figures apparaissent toujours parfaitement lisses.

- C'est un format XML. Quelle est sa structure (racine et enfants) ?

Vous pouvez vous amuser à rajouter un élément en vous inspirant des existants et regardez le résultat.

2.3. Format MathML (exercice optionnel)

Les fichiers MathML, extension `.mml`, définissent des équations mathématiques pouvant être incluses dans des documents HTML5 et DocBook. On en trouve dans Wikipedia. Par contre, pour les voir en tant que MathML sous Firefox, il faut installer l'extension Native MathML. En effet, comme la plupart des navigateurs n'affichent pas bien le MathML, Wikipedia propose automatiquement une image PNG à la place. Ou alors, il faut modifier nos préférences pour Wikipedia, voir cette page.

Télécharger le fichier `equation.mml`. Pour l'afficher, il suffit simplement de l'ouvrir dans le navigateur.

En ouvrant le document avec `XML Copy Editor` ou `geany`, vous allez constater que c'est assez complexe pour un résultat qui paraît simple. Tout est extrêmement décomposé et hiérarchisé. Le fichier montre plusieurs types d'éléments :

- `<mo>` pour un opérateur, avec une variante pour les parenthèses,
- `<mi>` pour un identifiant,
- `<mn>` pour un nombre,
- `<mrow>` pour des éléments à aligner horizontalement.

Ensuite, il y a des éléments pour indiquer la disposition, à vous de réfléchir :

- Quelle est la structure d'une expression type x^y ? Dans le même genre, il y a `<msub>` pour mettre un indice.

- Quelle est la structure d'une expression type $\frac{x}{y}$?

Avec ces connaissances, sauriez-vous créer en partant de rien un document MathML qui affiche $f(x) = x^2 - \frac{log(x)}{x+y}$?

2.4. Format GPX (exercice optionnel)

Ce format permet de représenter des itinéraires enregistrés par un capteur GPS.

Télécharger le fichier `trace.gpx`. Pour l'afficher, il faut utiliser un site web comme geoportail ou VisuGPX, dans ce dernier, cliquer sur « Visualiser une trace » et choisir le fichier à envoyer.

Regarder le contenu de ce fichier avec `geany`. La racine est l'élément `<gpx>`, son enfant principal est `<trk>`, il représente une « trace », c'est à dire une succession de points enregistrés. Les éléments `<trkpt>` représentent ces points.

- Remarquer le choix qui a été fait de placer les coordonnées géographiques en attribut, mais la date de passage et l'altitude en sous-éléments.
- Remarquer les espaces de nommage pour les extensions. Qu'est-ce qui définit le préfixe `gpxx` ?

3. Modélisation

On arrive maintenant à la partie création de fichiers XML. Le but est de représenter différents domaines le mieux possible. La norme XML offre de nombreuses choix :

- Choix des éléments et leur imbrication pour représenter les relations entre les informations,
- Utilisation de sous-éléments ou bien d'attributs pour représenter des informations,
- Espaces de nommages dans le cas d'éléments ou d'attributs ayant le même nom.

Les questions à se poser sont :

- Est-ce que les informations sont accessibles sans ambiguïté et de manière uniforme ?
- Est-ce extensible ? Pourra-t-on rajouter de nouveaux types d'informations à ce document ?

Un document XML représente une hiérarchie. Il vous faudra déterminer la racine, les branches dans cet arbre. Il faut que l'arbre soit suffisamment détaillé pour bien distinguer et identifier les informations, mais pas trop pour ne pas compliquer la recherche d'informations. Ce n'est que dans le TP3 (XPath) que vous apprendrez à extraire des informations d'un fichier XML.

Pour chacun des documents que vous allez créer, il vous est demandé de procéder à une vérification syntaxique à l'aide des outils installés à l'IUT :

- `xmlstarlet val -e document.xml`
- `xmllint --noout document.xml`

Si vous travaillez avec XML Copy Editor, le document sera automatiquement correct, mais vous pourrez forcer la vérification avec la touche `F2` ou le menu `XML`, `vérifier justesse de forme` ou encore le bouton V bleu de la barre d'outils.

La semaine prochaine, vous écrirez des fichiers DTD et des schémas permettant de vérifier la structuration et le contenu d'un document XML.

3.1. Cuisine

On s'intéresse aux recettes de cuisine, en général : entrées, plats, desserts. Modélisez les éléments nécessaires pour décrire trois ou quatre préparations culinaires simples : les ingrédients et les éléments nécessaires (four, casserole,...) . Ajoutez des informations comme le temps de préparation et le niveau de difficulté.

Vous devrez faire en sorte que la représentation de ces informations soit homogène : si vous placez le temps de préparation en tant qu'attribut dans l'une des recettes, il faudra qu'il en soit de même pour les autres. Vous aurez le choix entre attributs et éléments, faites comme vous voulez, connaissant les règles données en CM.

3.2. Terre

Voici un texte librement inspiré de Wikipédia. Essayez de modéliser les informations qu'il contient à l'aide d'XML. Vous allez devoir regrouper les informations le plus logiquement possible.

> La Terre est constituée de différentes couches. A l'extérieur, il y a la croûte terrestre, juste dessous, il y a le manteau supérieur qui fait environ 650 km d'épaisseur, puis plus bas, il y a le manteau inférieur qui fait 2200 km d'épaisseur. Le manteau supérieur est constitué d'olivine et de pyroxène, tandis que le manteau inférieur est constitué de pérovskite. La croûte terrestre est de deux sortes : la croûte océanique qui est en basalte de densité 3, épaisse de 6 km en moyenne contre 35 km pour la croûte continentale. La densité de cette dernière est de 2,7 car elle est en granite. Sous le manteau, il y a le noyau externe, de 200 km d'épaisseur et dessous encore il y a le noyau interne également appelé graine qui fait 1200 km de rayon. La graine est en fer, densité 13, tandis que le noyau externe est un mélange de fer et de nickel. La densité du manteau supérieur est d'environ 3,3 et celle du manteau inférieur atteint 6. Le noyau externe a une densité de 10.

Faites deux versions de ce document :

- `terre_attr.xml` dans lequel vous utilisez le plus possible d'attributs pour représenter les informations. Cela ne sera pas possible s'il y a plusieurs exemplaires de la même information, ex. composition, nom.
- `terre_elem.xml` dans lequel vous utilisez uniquement des sous-éléments pour représenter les informations.

3.3. Entités

Dans l'un des documents `terre.xml`, il y a les noms des couches telles que "croûte océanique", etc. Faites-en des entités et remplacez toutes les occurrences par des références. (Relire la fin du cours n°1).

Pour vérifier si les entités sont bien remplacées, faire :

- `xmllint --noent terre.xml`

3.4. Espaces de noms

Imaginons que vous ayiez à faire la jonction entre deux documents XML, l'un contient une liste de footballeurs et l'autre contient une liste de clubs dans lesquels ils ont joué[1].

Le document `footballeurs.xml` :

```xml
<?xml version="1.0" encoding="utf-8"?>
<footballeurs>
    <footballeur numero="1">
        <nom>Lilian Thuram</nom>
        <club id="C1" annees="1990-1996"/>
        <club id="C2" annees="2006-2008"/>
    </footballeur>
    <footballeur numero="2">
        <nom>Antoine Griezmann</nom>
        <club id="C2" annees="2019-2021"/>
    </footballeur>
</footballeurs>
```

Le document `clubs.xml` :

```xml
<?xml version="1.0" encoding="utf-8"?>
<clubs>
    <nom id="C1">AS Monaco</nom>
    <nom id="C2">FC Barcelone</nom>
</clubs>
```

Si on fait la fusion de ces deux documents en un seul, `footballeurs_clubs.xml`, indiquant le nom de chaque sportif et les clubs dans lequels chacun a travaillé, il y a un conflit sémantique sur la signification de l'élément `nom` :

```xml
<?xml version="1.0" encoding="utf-8"?>
<footballeurs>
    <footballeur numero="1">
        <nom>Lilian Thuram</nom>
        <nom id="C1" annees="1990-1996">AS Monaco</nom>
        <nom id="C2" annees="2006-2008">FC Barcelone</nom>
    </footballeur>
    <footballeur numero="2">
        <nom>Antoine Griezmann</nom>
        <nom id="C2" annees="2019-2021">FC Barcelone</nom>
    </footballeur>
</footballeurs>
```

[1] Désolé si vous n'aimez pas le foot. Comprenez que c'est seulement pour l'exercice.

On pourrait évidemment faire une hiérarchie pour les clubs, mais mettons qu'on ne veuille pas.

Rajoutez une notion de *namespace* dans ce dernier document pour résoudre le conflit. Il devrait y avoir un *namespace* global pour les éléments issus du document `footballeurs.xml` et un *namespace* avec préfixe pour les éléments issus de `club.xml`.

3.5. CDATA (exercice optionnel)

Voici un court document XML `emoticones.txt` à enregistrer en `.xml`. Vérifier s'il est correct syntaxiquement. Ce n'est pas le cas, alors faire en sorte, à l'aide de sections CDATA qu'il le devienne.

```
<?xml version="1.0" encoding="utf-8"?>
<emoticones>
    <sequence sens="sourire">:-)</sequence>
    <sequence sens="c'est la fête"><:-P</sequence>
    <sequence sens="je sors">--->[]</sequence>
</emoticones>
```

4. Travail à rendre

Vous avez travaillé dans le dossier `tp1`. Remontez au dessus avec le navigateur de fichiers. Cliquez droit sur le dossier `tp1`, choisissez `Compresser...`, cliquez sur `Créer`. Ça va créer une archive `tp1.zip`. Déposez cette archive sur Moodle, dans la page de cours dédiée L4IN121T Formats et traitements de données internet.