Compte rendu projet professionnel « TEI2JSON »

EL HELOU Myriam
(Numéro d'étudiant – 10468326)

BOUHOUCHE Sami
(Numéro d'étudiant – 10468326)

UFR LLASIC
Département des Sciences du Langage et Didactique des Langues

Master 2 Mention Sciences du Langage Parcours Industries de la Langue
UE : Projet professionnel

Commanditaire : Arnaud BEY
Enseignant responsable : Thomas LEBARBE

Année universitaire
2018 – 2019

Sommaire

Rappel des caractéristiques du projet

Les parties prenantes :

Le commanditaire :

- Arnaud BEY : ingénieur au département Litt&Art de l'Université Grenoble Alpes.

L'équipe Projet :

- Myriam EL HELOU
- Sami BOUHOUCHE

La demande :

Le commanditaire est amené dans le cadre de ses activités à encoder des documents en TEI. Il est possible de récupérer les caractéristiques des balises de la TEI sous différents formats depuis l'outil TEI-ROMA qui se trouve sur le site officiel de la TEI. Les formats possibles sont les suivants :

- relax NG compact
- relax NG XML
- ISO Schematron
- Schematron
- W3C Schema
- DTD

Le format privilégié par notre commanditaire est le relax NG avec une syntaxe XML. Cependant, à l'image des autres formats de sorties proposées par ROMA, celui-ci est complexe et difficilement compréhensible sans une certaine connaissance des langages du web sémantique. Ainsi, afin de pallier à cela, notre commanditaire nous a demandé de mettre en place **un outil permettant de convertir les documents .rng issus de ROMA en documents au format JSON.**

La proposition :

La proposition sur laquelle le commanditaire et l'équipe projet se sont mis d'accord consiste à mettre en place un code qui convertit un fichier rng en un fichier JSON. Ce dernier récupérera les caractéristiques suivantes :

- Les balises TEI correspondant à un certain module
- Les descriptions/documentations des balises TEI
- Les attributs ainsi que leurs valeurs
- Les enfants autorisés pour chaque balise

Et les écrira dans un fichier au format JSON.

Cet outil sera développé en python 3.5 et utilisera le module beautifulsoup qui permet de parser -entre autres- du XML.

Résultats attendus (vu dans le cahier des charges)

Exemple de fichier en entrée :

```
      <ref name="tei_att.ranging.attribute.atMost"/>
      <ref name="tei_att.ranging.attribute.min"/>
      <ref name="tei_att.ranging.attribute.max"/>
      <ref name="tei_att.ranging.attribute.confidence"/>
   </define>
   <define name="tei_att.ranging.attribute.atLeast">
      <optional>
         <attribute name="atLeast">
            <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0">gives a minimum estimated value for the
            approximate measurement.</a:documentation>
            <choice>
               <data type="double"/>
               <data type="token">
                  <param name="pattern">(\-?[\d]+/\-?[\d]+)</param>
               </data>
               <data type="decimal"/>
            </choice>
         </attribute>
      </optional>
   </define>
   <define name="tei_att.ranging.attribute.atMost">
      <optional>
         <attribute name="atMost">
            <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0">gives a maximum estimated value for the
            approximate measurement.</a:documentation>
            <choice>
               <data type="double"/>
               <data type="token">
                  <param name="pattern">(\-?[\d]+/\-?[\d]+)</param>
               </data>
               <data type="decimal"/>
            </choice>
         </attribute>
      </optional>
   </define>
   <define name="tei_att.ranging.attribute.min">
      <optional>
         <attribute name="min">
```

Exemple théorique de la sortie:

```
{
   "elements":[{
      "tag":"recording ",
      "self_closed":" ",
      "doc" :
      {
         "en": "(recording event) provides details of an audio or video recording event used as the source of a spoken text,
         either directly or from a public broadcast. [8.2. Documenting the Source of Transcribed Speech 15.3.2. Declarable Elements]"
      },
      "attributs":[{
         "key":"type ",
         "type":"String",
         "required":false,
         "doc":
            {
            "en":"the kind of recording."
            },
         "value":[]
      }
      ],
      "childrens":[]

}
]
}
```

Explication de la réalisation

Les caractéristiques de la solution :

Librairie/ressources utilisées

Librairies à installer :

- BeautifulSoup : Beautiful Soup est une bibliothèque Python permettant d'extraire des données de fichiers HTML et XML. Cette librairie devrait fonctionner de la même manière dans Python 2.7 et Python 3.2 et plus.

Librairies internes à python :

- JSON : JSON est une bibliothèque Python permettant de former à partir d'objets des fichiers valides en format JSON. Cette bibliothèque devrait être compatible avec les versions Python 2.6 et plus.
- re : Re est une bibliothèque Python qui permet d'utiliser les expressions régulières. Cette bibliothèque devrait être compatible avec les versions Python 2.6 et plus.
- sys : ce module donne accès à certaines variables utilisées ou gérées par le terminal. Il nous sera utile pour récupérer le nom du fichier à traiter.

Le développement :

Décomposition du code :

Le code a été développé en 4 temps :

1. Récupération du nom de la balise ainsi que de sa documentation (en français lorsque cette dernière est disponible)
2. Récupération des attributs, ainsi que leurs valeurs :
    - Le nom de l'attribut.
    - La documentation.
    - Le type de l'attribut (String ou Enumerated)
    - L'attribut est-il obligatoire ou non ? (« required », toujours « false » à la demande de notre commanditaire).
    - Les valeurs de l'attribut
3. Récupération des enfants.
4. Assemblage de toutes les informations et écriture de celles-ci dans le JSON

Explication du code :

1. **Récupération du nom et de la documentation :** En utilisant les fonctionnalités de la librairie beautifulsoup, nous avons recherché la balise « element » puis pour chaque balise nous avons recherché le nom de l'élément ainsi que sa documentation. Nous avons par la suite initialisé les variables « tag » et « documentation » avec les valeurs récupérées.
   Le nom et la documentation se trouvent directement dans le contenu de la balise « element » du fichier rng, ainsi nous avons mutualisé leur traitement.

2. **Récupération des attributs ainsi que leurs valeurs :** Cette fonctionnalité a été développée séparément dans le fichier « recup_attributes.py » et par la suite importée dans le code principal « TEI2JSON.py ». Elle permet pour chaque attribut de récupérer les valeurs citées dans la partie précédente *(Décomposition du code)*.

Nous distinguons 2 cas de récupération : soit les attributs se trouvent directement dans la balise « element », soit la balise « element » contient une référence vers une liste d'attributs qu'admet l'élément. Cette particularité nous a obligés à traiter chaque cas d'une manière différente :

- Pour le premier cas : Nous avons recherché la balise « attribute » dans la balise « element » puis nous avons appliqué le code chargé de récupérer les caractéristiques de l'attribut.
- Pour le deuxième cas : Nous avons récupéré le nom de la référence trouvé dans l'élément puis nous avons recherché sa définition dans le document. Il faut noter que les références trouvées peuvent renvoyer soit à un seul attribut soit à un groupe d'attributs. Dans le cas où les références renvoient à un nouveau groupe d'attributs, nous avons refait la même procédure.

3. **Récupération des enfants :** Dans un premier temps, nous avons mis en place une approche basée sur la description des éléments présente sur le site officiel de la TEI. Nous avons utilisé les fonctionnalités de beautifulsoup afin de rechercher l'élément et d'en aspirer les enfants. Par la suite, on comparait les enfants obtenus avec les éléments présents dans le fichier .rng afin de ne garder que les éléments qui se trouvent dans celui-ci. Cependant cette approche a rapidement montré ses limites. Suite à une panne des serveurs du site de la TEI, note code ne fonctionnait plus. Ainsi nous avons décidé de n'utiliser que les informations présentes dans le .rng afin que notre outil puisse récupérer correctement les enfants.
   La solution retenue consiste à récupérer récursivement les enfants. Cette solution se trouve dans le module « recup_children.py ».
   Avant de commencer à coder, nous avons extrait les deux conditions d'arrêt :
   - Le premier cas : lorsque nous rencontrons une balise <notAllowed>. Dans ce cas, nous ignorons cet élément.
   - Le deuxième cas : lorsque nous nous retrouvons avec la balise <element> qui contient l'enfant à rajouter à liste des enfants de l'élément.

   Dans le dernier cas, où nous avons des références, nous les récupérons toutes sauf celles qui débutent par « tei_att » et nous appliquons la récursivité dessus.

4. **Assemblage et écriture :** Toutes les informations récupérées, dans les étapes précédentes, ont été regroupées dans un tableau associatif. La fonction json.dumps() permet de représenter le contenu du tableau sous format JSON.

Résultats obtenus

Exemple du contenu du fichier d'entrée (.rng)

```
<define name="tei_said">
   <element name="said">
      <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0">(speech or thought) indicates passages thought
      or spoken aloud, whether explicitly indicated in the source or not, whether directly or indirectly reported, whether by real
      people or fictional characters. [3.3.3. Quotation]</a:documentation>
      <ref name="tei_macro.specialPara"/>
      <ref name="tei_att.global.attributes"/>
      <ref name="tei_att.ascribed.directed.attributes"/>
      <optional>
         <attribute name="aloud">
            <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0">may be used to indicate whether the
            quoted matter is regarded as having been vocalized or signed.</a:documentation>
            <choice>
               <data type="boolean"/>
               <choice>
                  <value>unknown</value>
                  <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"/>
                  <value>inapplicable</value>
                  <a:documentation xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"/>
               </choice>
            </choice>
         </attribute>
      </optional>
      <optional>
         <attribute xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
                    name="direct"
                    a:defaultValue="true">
            <a:documentation>may be used to indicate whether the quoted matter is regarded as direct or indirect speech.</a:
            documentation>
            <choice>
               <data type="boolean"/>
               <choice>
                  <value>unknown</value>
                  <a:documentation/>
                  <value>inapplicable</value>
                  <a:documentation/>
               </choice>
            </choice>
         </attribute>
      </optional>
      <empty/>
   </element>
```

Exemple du contenu du fichier de sortie (.json)


```
"tag": "said",
"documentation": "(speech or thought) indicates passages thought or spoken aloud, whether explicitly indicated in the source or not
"attributes": [
    {
        "key": "rend",
        "type": "string",
        "required": false,
        "documentation": "(rendition) indicates how the element in question was rendered or presented in the source text.",
        "values": []
    },
    {
        "key": "style",
        "type": "string",
        "required": false,
        "documentation": "contains an expression in some formal style definition language which defines the rendering or presentati
        "values": []
    },
    {
        "key": "rendition",
        "type": "string",
        "required": false,
        "documentation": "points to a description of the rendering or presentation used for this element in the source text.",
        "values": []
    },
    {
        "key": "cert",
        "type": "enumerated",
        "required": false,
        "documentation": "(certainty) signifies the degree of certainty associated with the intervention or interpretation.",
        "values": [
            "high",
            "medium",
            "low",
            "unknown"
        ]
    },
```

Nous pouvons observer quelques différences entre les résultats attendus et les résultats obtenus. Ces modifications ont été effectuées suite à la demande du commanditaire. Nous pouvons citer:

- La suppression de la caractéristique « self-closed »
- La suppression de la documentation en deux langues et la conservation de la documentation en français.

Répartition des tâches

La répartition des tâches que nous avons prévu dans notre Trello (https://trello.com/b/kiVvRajs/projet-professionnel-tei2json) a été respectée.

Cette répartition est également représentée sous forme d'un récapitulatif des grandes tâches dans le tableau ci-dessous :

Tâche	Myriam	Sami
Récupérer les éléments	X	X
Récupérer la documentation	X	X
Récupérer les attributs ainsi que leurs valeurs	X	
Récupérer les enfants		X
Faire la sortie finale JSON	X	X
Débuggage	X	X
Écriture des documents	X	X

Les délais de la majorité des tâches ont été respecté sauf pour la tâche de récupération des enfants. Ce retard a été causé par la panne des serveurs du site de la TEI, ce qui nous a obligé a changé le code de cette partie.