Un peu de XPath, un peu de XSLT

2011-05

Dans ce petit exercice et le suivant, on prend comme exemple une version numérisée d'un célèbre sonnet de DuBellay. Vous trouverez une version de ce sonnet déjà balisée dans le fichier duBellay.xml — ouvrez-le d'abord avec Oxygen.

1 XPath

1.1 Navigation avec XPath

oXygen est très pratique pour voir un peu ce que l'utilisation de XPath seul peut apporter à un processus d'exploration, d'appropriation ou de vérification de document TEI. La ligne de saisie "XPath" que l'on voit a gauche en haut de la fenêtre, au-dessous des barres d'outils oXygen, permet de saisir des expressions XPath. Après avoir tapé sur la touche Entrée au clavier, oXygen évalue l'expression et retourne, dans la fenêtre qui fait toute la largeur de l'écran en bas, une liste de(s) résultat(s) – un ou plusieurs ensembles de nœuds, en donnant pour chaque nœud résultat l'expression XPath qui permet de l'atteindre. Si on clique avec la souris sur une des lignes de résultat, oXygen montre le nœud résultat sélectionné dans son contexte TEI au sein de la fenêtre de l'éditeur.

Voici une petite liste de requêtes que vous devriez essayer d'exprimer au moyen du langage XPath :

1. quel est l'élément racine du fichier XML ?
2. quel est le titre donné à ce fichier ?
3. qui est le responsable de la création du fichier ?
4. quelle est l'origine de la version encodée ?
5. quelle est la date de la révision la plus récente du fichier ?
6. combien y a-t-il de strophes (`<lg>`) dans le sonnet ?
7. combien y a-t-il de segments de texte dont la graphie est considérée incorrecte aujourd'hui ?
8. combien y a-t-il de vers (`<l>`) qui contiennent au moins un tel segment ?
9. quel est le contenu textuel du premier vers de chaque strophe ? et du dernier?
10. Quels nœuds texte contiennent la chaîne de caractères "jamais"

On va faire ces exercices ensemble. Des réponses sont proposées dans la section qui suit : ne les regardez pas si vous préférez tester vos compétences !

Nota : on utilise ici XPath 2.0 (à cause de l'espace de noms par défaut).

1.2 Exercice XPath corrigé

Quel est l'élément racine du fichier XML ? Expression XPath à écrire :

> `/*` [le nœud document est désigné par le slash ; la 2^{e} étape désigne les enfants de type élément de ce nœud]
>
> Réponse : TEI

A l'Ambicieux,

ET AVARE ENNEMY

DES BONNES LETTRES.

Sonnet.

Serf de Faueur, Eſclaue d'Auarice,
Tu n'heus iamais ſur toymeſmes pouuoir,
Et ie me veux d'vn tel Maitre pourueoir,
Que l'Eſprit libre en plaiſir ſe nouriſſe.
L'Air, la Fortune, & l'humaine Police
Ont en leurs Mains ton malheureux Auoir.
Le Iuge auare icy n'a rien à voir.
Ny les troys Seurs, ny du Tens la malice.
Regarde donc qui eſt plus ſoubaitable
L'ayſe, ou l'ennuy, le certain, ou l'inſtable.
Quand à l'honneur, i'eſpere eſtre immortel:
Car vn cler Nom ſoubz Mort iamais ne tumbe.
Le tien obſcur ne te promet rien tel.
Ainſi, tous deux ſerez ſoubz meſme Tumbe.

CAELO MVSA BEAT.

Quel est le titre donné au fichier ? Expression XPath à écrire :

`/TEI/teiHeader/fileDesc/titleStmt/title` ou encore (entre autres possibilités) : `//titleStmt/title`

Réponse : On voit les deux nœuds `<title>` qui répondent à la question. Si on voulait obtenir le titre proprement dit (sans le complément de titre), il faudrait connaître précisément le modèle auquel obéit ce document, et demander le nœud `<title>` en première position (//titleStmt/title[1]), ou de choisir le nœud `<title>` dont l'attribut *@type* porte une valeur appropriée (`//titleStmt/title[@type='main']`).

Qui est le responsable de la création du fichier ? Expression XPath à écrire :

`//titleStmt/respStmt/name`

Réponse : 1 seul nœud ici.

Quelle est l'origine de la version encodée ? Expression XPath à écrire :

`//sourceDesc/*` ou, plus finement, `/TEI/fileDesc/sourceDesc/*`. Sans le * vous ne recevrez que l'élément de groupement `<sourceDesc>`, sans ses composants.

Quelle est la date de la révision la plus récente du fichier ? Expression XPath à écrire :

`revisionDesc/change[1]/@when`. Par défaut, le premier `<change>` devrait être le plus récent, et son attribut *@when* donnera sa date dans un format normalisé.

Réponse : 2011-05-23

Combien y a-t-il de strophes (`<lg>`) dans le sonnet ? Expression XPath à écrire :

`count(//lg)`. On se sert de la fonction `count()`.

Réponse : 4

Combien y a-t-il de `<orig>`, i.e. des segments de texte dont la graphie est consideré incorrecte
Expression XPath à écrire :

`count(/TEI/text//orig)`. Enlevez le `count()` si vous préfériez les voir…

Réponse : 16

Combien y a-t-il de vers (`<l>`) qui contient au moins un tel segment? Expression XPath à écrire :

`count(//l[orig])`

Réponse : 7

Et combien y a-t-il de `<orig>` qui sont contenu directement par un `<l>` ?
Expression XPath à écrire :

`count(//l/orig)`

Réponse : 6

Quel est le contenu textuel du premier vers de chaque strophe ? et du dernier ?
Expression XPath à écrire :

`//lg/l[1]//text()`

Pour le dernier, c'est un peu moins evident… `//lg/l[position()=last()]//text()`

Quels nœuds texte contiennent la chaîne de caractères "jamais" ? Expression XPath à écrire :

`//text()[contains(., 'jamais')]`

Réponse : la liste de réponses contient 2 nœuds

2 XSLT

Dans ce deuxième petit exercice, nous allons faire des expériences avec XSLT sous contrôle d'oXygen. On va travailler avec cette version du même sonnet de DuBellay qui a été (partiellement) enrichie avec des balises `<choice>`.

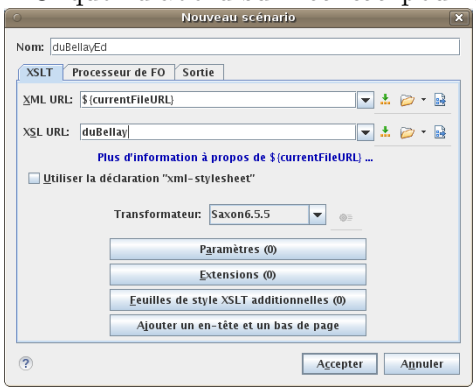
2.1 Transformation en HTML

L'application classique de XSLT est de transformer un document TEI XML en HTML pour le visualiser dans un navigateur web. On commence donc par là.

D'abord il faut mettre en place une liaison entre le fichier XML d'entrée, le fichier HTML de sortie, et le fichier XSLT qui va contrôler la transformation. Cette liaison s'appelle un *scénario de transformation* et on la configure dans oXygen de la manière suivante:

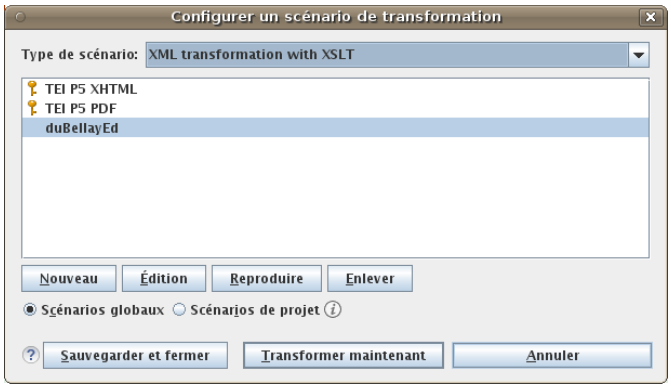
1. Ouvrez Oxygen. Sélectionnez Ouvrir sur le menu Fichier et ouvrez le fichier duBellay.xml.
2. Sur le menu Document sélectionnez Transformation, et ensuite Configurer un scénario de transformation ou tapez `ctrl-maj-C`, ou bien cliquez sur le bouton à droite du triangle rouge sur la barre d'outils.

Cliquez d'abord sur nouveau pour ouvrir la boîte de dialogue Nouveau Scénario.



- Tapez `duBellay.xsl` dans le champ XSL URL.
- Cliquez sur l'onglet `Sortie`
- Tapez `output.html` dans le champ `Enregistrer sous`
- Cochez la case `Ouvrir dans un navigateur`
- Cliquez sur `Accepter`

Vous avez maintenant créé un nouveau scénario de transformation, qui s'appelle duBellayEd.



- Cliquez sur `Transformer maintenant`

- En bas de l'écran, le message `Transformation réussie` apparaît…
- .. et après un bref délai, votre navigateur devrait s'ouvrir pour afficher le fichier output.html que vous venez de créer.
- C'est joli, hein ? Peut-être y a-t-il encore un peu de travail à faire…

2.2 Une feuille de style XSLT

Une feuille de style XSLT est un document XML. Il peut donc être édité avec Oxygen.

- Allez de nouveau sur Fichier - Ouvrir. Ouvrez le fichier duBellay.xsl dans votre dossier de travail.
- Ce fichier XSLT contient un seul template, qui correspond à l'élément `<TEI>`, l'élément racine de notre document. Son effet sera de produire les balises `<html>` et `</html>`, avec entre les deux le résultat du `<xsl:apply-templates>`
- Ce dernier va essayer d'appliquer tous les templates disponibles… il n'y en a pas, donc il ne va produire que le texte du document.
- … et en effet, si vous revenez sur votre navigateur et regardez la source du fichier, c'est ce qu'il a fait.

Au travail ! D'abord, nous allons extraire de l'en-tête un titre pour le document HTML, et supprimer le reste du TEI Header.

- Après le `<html>`, tapez `<head>`
- oXygen ajoute la balise fermante. C'est bien. Continuez donc. Ajoutez `<title>` à l'intérieur de l'élément `<head>`. Pour trouver le titre du document il faut naviguer de la racine du document (l'élément TEI) jusqu'à l'élément `<title>`, qui se trouve dans le `<titleStmt>`, dans le `<fileDesc>`, dans le `<teiHeader>`. Cela s'effectue avec le *XPath* suivant.
- Tapez `<xsl:value-of select="teiHeader/fileDesc/titleStmt/title"/>`.
- Modifiez la balise `<xsl:apply-templates>`, en ajoutant `select="text"` et entourez-la d'un élément HTML `<body>`
- Cliquez sur le bouton Indentation. Votre feuille de style maintenant devrait ressembler à ceci :

```
1  <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
2    xpath-default-namespace="http://www.tei-c.org/ns/1.0" version="2.0">
3
4    <xsl:template match="TEI">
5      <html>
6        <head>
7          <title>
8            <xsl:value-of select="teiHeader/fileDesc/titleStmt/title"/>
9          </title>
10       </head>
11       <body>
12         <xsl:apply-templates select="text"/>
13       </body>
14     </html>
15   </xsl:template>
16
17 </xsl:stylesheet>
18
```

- Cliquez sur l'icône de la disquette (ou tapez ctrl-s) pour enregistrer les modifications que vous venez de faire.
- Cliquez sur l'onglet duBellayEd.xml pour revenir sur votre document XML.
- Cliquez sur l'icône triangle-rouge pour relancer la transformation (ou tapez CTRL-maj-T)
- Voyons ce que cela donne : cette fois, on ne voit que le `<text>` de notre document. On fait des progrès !

Maintenant, on va ajouter des templates.

- Revenez sur la feuille de style.
- Ajoutez
  - un template pour `<l>`, qui va ajouter une balise `<br/>` après chaque vers
  - un template pour `<head>`, qui va l'entourer de `<h2>`
  - un template pour `<lg>`, qui va l'entourer de `<p>`, et en plus le préfixer du nombre de chaque strophe.
- Voici les templates requis… essayez de comprendre par vous-même le fonctionnement de chacun d'entre eux.

```
17  <xsl:template match="l">
18    <xsl:apply-templates/><br/>
19  </xsl:template>
20
21  <xsl:template match="head">
22    <h2><xsl:apply-templates/></h2>
23  </xsl:template>
24
25  <xsl:template match="lg">
26    <xsl:number/>
27    <p><xsl:apply-templates/></p>
28  </xsl:template>
29
```

2.3 Traitement des `<choice>`

Est-ce que le XSLT peut nous aider à mieux traiter ces éléments `<choice>`? Bien sur, oui. D'abord, on pourrait simplement ajouter un template qui va supprimer les `<orig>` (ou les `<reg>` si vous le préfériez).

Tout ce qu'il faut, c'est ajouter un template comme ceci : `<xsl:template match="orig"/>`. Essayez-le. Est-ce que vous comprenez comment cela fonctionne ? Si oui, vous avez bien compris les principes de XSLT !

2.4 Pour aller plus loin…

- Créez un paramètre modernisation avec `<xsl:variable>`, avec comme valeur "oui" ou "non", selon votre préférence. (Attention de ne pas introduire des blancs autour du mot !). Ceci va contrôler la suppression ou bien des `<reg>`, ou bien des `<orig>` selon la valeur qu'on lui donne. Par exemple, s'il aura la valeur `oui` on supprimera les `<orig>`.

- Ajouter un template pour l'élément `<choice>`, qui contiendra un `<xsl:choose>` pour tester la valeur de ce paramètre
- Les experts en HTML peuvent proposer d'autre mises en jours éventuelles pour les `<choice>`, sous contrôle d'autre valeurs du paramètre…

Qu'est-ce qu'on ferait pour n'afficher qu'une partie du texte? Modifiez votre feuille de style pour n'afficher que le vers initial de chaque strophe. Ensuite, vous pourriez les trier par ordre alphabétique….

Vous trouverez dans le fichier duBellayCorr.xsl une version corrigée de cet exercice… ne le regardez pas tout de suite !