Séance 2

Les règles de base de XSL

Mettre en place une règle

- Pour donner des instructions XSL, on utilise des templates/règles grâce à l'élément :

```
<xsl:template>
```

- On utilise l'attribut match pour sélectionner un élément de l'arbre XML :

```
<xsl:template match="mon_element_xml">
```

On peut ajouter dans la règle du texte,

```
<xsl:template match="mon_element_xml">
    Ici, il y avait mon élément
</xsl:template>
```

mais aussi des balises, ou un motif qui permet de récupérer certains éléments de l'arbre XML soit tels quels, soit transformés ou triés par des fonctions XSL ou Xpath.

```
<xsl:template match="mon_element_xml">
    <xsl:text>Ici, il y avait mon élément</xsl:text>
</xsl:template>
```

Éléments et attributs XSL

Éléments

Méthode n°1 :

```
<xsl:template match="mon_element_xml">
   <p>Ici, il y avait mon élément</p>
</xsl:template>
```

Méthode n°2 :

```
<xsl:template match="mon_element_xml">
<xsl:element name="p">Ici, il y avait mon élément</xsl:element>
</xsl:template>
```

Attributs

Méthode n°1 :

```
<xsl:template match="mon_element_xml">
  <p type="valeur_attribut">Ici,
    il y avait mon élément</p>
</xsl:template>
```

ou

```
<xsl:template match="mon_element_xml">
        <p type="{./chemin_Xpath}">Ici,
          il y avait mon élément</p>
</xsl:template>
```

Méthode n°2 :

```
<xsl:template match="mon_element_xml">
   <xsl:element name="p">
        <xsl:attribute name="type">
             <xsl:text>valeur_attribut</xsl:text>
        </xsl:attribute>
        <xsl:text>Ici, il y avait mon élément</xsl:text>
    </xsl:element>
</xsl:template>
```

Apply-templates

Cette règle indique que les règles définies dans l'XSL doivent être appliquées aux éléments enfants de l'élément sélectionné par la règle.

Exemple

```
<xsl:template match="mon_element_xml">
     <xsl:element name="p">
          <xsl:apply-templates/>
     </xsl:element>
</xsl:template>
```

Copy et copy-of

Copy

« L'élément XSL copy copie le nœud courant du document source vers le document de sortie. Il ne copie que le nœud lui-même. Cependant il ne copie pas ses enfants et ses attributs. »

Elliotte Rusty Harold, W. Scott Means, Philippe Ensarguet[et al.], *XML en concentré*, Paris, O'Reilly, 2005, p. 525.

```
<xsl:template match="mon_element_xml">
        <xsl:copy/>
</xsl:template>
```

Copy-of

« L'instruction xsl:copy-of insère le fragment d'arbre résultat identifié par l'attribut select dans le document de sortie. Cette instruction copie les nœuds spécifiques sélectionnés par l'expression et tous leurs enfants, attributs, espaces de noms et descendants. C'est en cela qu'il diffère de xsl:copy.»

Elliotte Rusty Harold, W. Scott Means, Philippe Ensarguet[et al.], *XML en concentré*, Paris, O'Reilly, 2005, p. 525.

```
<xsl:template match="mon_element_xml">
        <xsl:copy-of/>
</xsl:template>
```

Value-of

« L'élément xsl:value-of calcule la valeur textuelle d'une expression Xpath et l'insère dans l'arbre résultat. »

Elliotte Rusty Harold, W. Scott Means, Philippe Ensarguet[et al.], *XML en concentré*, Paris, O'Reilly, 2005, p. 543.

```
<xsl:template match="mon_element_xml">
        <xsl:value-of select="chemin_Xpath"/>
</xsl:template>
```

Numéroter automatiquement des vers

L'élément **xsl:number** est utilisé pour produire un nombre formaté dans l'arbre résultat.

L'élément possède plusieurs attributs :

- L'attribut *count* est un motif qui spécifie les nœuds à comptabiliser dans les nœuds spécifiés.
- L'attribut *level* défini le nombre de niveaux "any / multiple / single"
- L'attribut *format* permet de paramétrer le format de numérotation

```
<xsl:template match="mon_element_xml">
    <xsl:number count="element_a_compter" level="" format=""/>
</xsl:template>
```

Exercice

À l’aide des éléments vus pendant la séance, reproduire l'intégralité du fichier TEI de Verlaine en numérotant automatiquement les vers et les strophes. Essayer de proposer plusieurs formats et plusieurs types de numérotation des vers.