Séance 3 : Allez plus loin dans l'usage des règles de base

apply-templates et l'attribut select

```
<xsl:template match="mon_element_TEI">
   <xsl:apply-templates select="sous_elements_xml"/>
</xsl:template>
```

apply-templates et l'attribut mode

« Parfois le même contenu en entrée doit apparaître plusieurs fois dans le document de sortie, formaté selon un modèle différent à chaque fois. [...] Les éléments *xsl:appy-templates* et *xsl:template* peuvent avoir un attribut mode optionnel qui associe différentes règles à différents usages. L'attribut mode d'un élément *xsl:template* identifie dans quel mode cette règle-modèle devait être activée. Un élément *xsl:apply-template* avec un attribut mode n'active que la règle modèle avec l'attribut mode correspondant. »
Elliotte Rusty Harold, W. Scott Means, Philippe Ensarguet[et al.], *XML en concentré*, Paris, O'Reilly, 2005, p. 171.

```
 <xsl:template match="element">
      <xsl:apply-templates mode="nom_mode"/>
</xsl:template>

<xsl:template match="sous_element" mode="nom_mode">
      Règles correspondant au mode
</xsl:template>
```

Exercice

À l'aide des éléments que nous avons vus dans les séances 2 et 3, créer un nouveau fichier XML à partir de sainteEulalie.xml. Le fichier devra contenir les éléments suivants :

- **Etape n°1**
  - Un header identique au document source
  - un body avec un titre (head)
- **Etape n°2**
  - deux groupes de lg distincts :
    - un lg @type="orig" qui donnera le contenu de la cantilène organisé en vers numérotés de manière continue et présentant le texte avec la graphie du manuscrit source.
    - un lg @type="reg" qui donnera le contenu de la cantilène organisé en vers numérotés et présentant le texte avec une graphie normalisée.

La création de variables

« L'élément *xsl:variable* attache une valeur de n'importe quel type (chaîne de caractère, nombre, ensemble de nœuds, etc.) à un nom. Cette variable peut être déréférencée par la suite en utilisant la forme $nom dans une expression Xpath. »
Elliotte Rusty Harold, W. Scott Means, Philippe Ensarguet[et al.], *XML en concentré*, Paris, O'Reilly, 2005, p. 544.

```
<xsl:variable name="nom_variable" select="Xpath"/>
ou
<xsl:variable name="variable"> Règles internes à la variable
</xsl:variable>
```

La variable est appelée par la suite par $nom_variable

```
<xsl:copy-of select="$variable"/>
<xsl:value-of select="$variable"/>
```

Fontions Xpath et traitement des chaînes de caractères

- Concat : permet de souder ensemble des chaînes de caractères
  - concat(string,string,...)

    ex : `<xsl:value-of select= ‘concat(‘element1’, ‘element2‘, ‘element3’)`
- Replace : permet de modifier une chaîne de caractères
  - replace(string,pattern,replace)

    ex : `<xsl:value-of select= ‘replace(element_a_modifier, ‘caracteres_a_remplacer‘, ‘caractere_de_remplacement’)`

- Upper-case : permet de passer en majuscule une chaîne de caractères
    - upper-case(string)
- Lower-case : permet de passer en minuscule une chaîne de caractères.
    - lower-case(string)

Pour voir plus de fonctions Xpath :
https://www.w3schools.com/xml/xsl_functions.asp

Exercice

Transformer le fichier XML mon rêve familier en fichier HTML à afficher.

1. Créer un nouveau fichier XSLT
2. Définir le namespace tei par défaut dans votre Xpath.
3. Paramétrer HTML comme format de sortie avec un encodage utf-8 à l'aide de xsl:output @method @encodage.
4. Créer une structure HTML d'accueil des données XML :
    - Ajouter dans un titre (h1) avant le poème contenant le titre, comprenant le « title », « écrit par », « author ».
    - Structurer votre poème à l'aide de ul pour les tercets et les quatrains et li pour les vers

5. Remplacer dans le titre « Mon » par « Un »
6. Trouver un moyen de numéroter les vers dans le HTML pour que le numéro n'apparaisse qu'au passage de la souris.
7. Bonus : ajouter à la fin du poème, la ligne "Ce poème contient [x=nb de vers] vers", puis remplacer « poème » par la valeur de @type du lg qui contient l'intégralité du poème.