Modélisation et interopérabilité : Semaine 46, cours 8

Benoît Valiron <benoit.valiron@monoidal.net>

http://inf356.monoidal.net/

XPath

- Chemin absolu, chemin relatif vis à vis d'un noeud contextuel
  - `eltenfant`
  - `@att`
  - `text()`
  - `elt1|elt2`
  - `//`
  - `..`
  - `.`
  - `*`
- Notion d'axes : `child, descendant, ancestor, siblings`
- Types d'expressions : booléen, nombre, chaine de caractères, ensemble de noeuds

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

XPath : /liste/livre/titre

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

XPath : /liste/livre/auteur

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

XPath : /liste/livre/auteur|/liste/livre/titre

Autre moyen : /liste/livre/*[name()='auteur' or name()='titre']

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

XPath : /liste/livre[@genre='web']

XSLT :
Transformations XML

XSLT

- V. 1.0 : recommandation W3C depuis 1999
- V. 2.0 : recommandation depuis 2007

  (comme pour XPath...)
- Description d'une transformation d'un document XML.
- Espace de noms : `http://www.w3.org/1999/XSL/Transform`
- Extension : `.xsl`

```
<stylesheet version='1.0'
     xmlns='http://www.w3.org/1999/XSL/Transform'>

  <output method="xml"/>

  <template match='...'>
   …
  </template>

</stylesheet>
```

- Format général pour le document.
- `xml` Peut être aussi `html`, ou `text`. Le comportement est différent.

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Livre trouvé
Livre trouvé
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
    Livre trouvé
  </xsl:template>

</xsl:stylesheet>
```

XSLT : Généralités

- Basé sur des structures appelées modèles (template).
- Document source –(feuille de style)→ document résultat.
- Un modèle décrit une règle, c'est-à-dire une action à effectuer pour chaque noeud décrit par l'expression XPath contenue dans l'attribut `match`
- Il y a un “template” par défaut qui affiche tout le texte d'un noeud donné.
- L'attribut `priority` prend une valeur décimale.
- Par défaut, le modèle avec le match le plus précis a la priorité maximale.

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Réponse 1
Réponse 1
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre” priority=”0.8”>
    Réponse 1
  </xsl:template>

  <xsl:template match=”/liste/livre” priority=”0.2”>
    Réponse 2
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Réponse 2
Réponse 2
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre” priority=”0.2”>
    Réponse 1
  </xsl:template>

  <xsl:template match=”/liste/livre” priority=”0.8”>
    Réponse 2
  </xsl:template>

</xsl:stylesheet>
```

Exemple : par défaut

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
XML
Chagnon
2007


Quantum Computation
Nielsen
Chuang
2001
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

</xsl:stylesheet>
```

On va voir plus loin comment ça marche...

<value-of select="..." />

- Rends une expression XPath de type texte

  ou la transforme en texte :

  - Du texte : Ok
  - Un noeud → rend le texte du noeud
  - Un ensemble de noeuds → utilise le premier noeud
- Remplace le <...> par le texte en question

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Titre : XML
Titre : Quantum Computation
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
    Titre : <xsl:value-of select=”titre” />
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Livre web
Livre quantique
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
     Livre <xsl:value-of select=”@genre” />
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Livre "
XML
Chagnon
2007
"
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste”>
     Livre “<xsl:value-of select=”livre” />”
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Auteur : Chagnon
Auteur : Nielsen
```
???

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
     Livre “<xsl:value-of select=”auteur” />”
  </xsl:template>

</xsl:stylesheet>
```

`<for-each select="..."> exp </for-each>`

- Prends une expression XPath de type ensemble de noeuds et pour chacun :
  - Place le noeud contextuel dessus
  - Évalue `exp` dans chacun des cas
- Produit donc autant de fois `exp` qu'il y a de noeuds dans l'ensemble.

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Auteur : Chagnon
Auteur : NielsenChuang
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
     Auteur :
     <xsl:for-each select=”auteur”>
       <xsl:value-of select=”text()” />
     </xsl:for-each>
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Auteur : Chagnon
Auteur : Nielsen Chuang
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
     Auteur :
     <xsl:for-each select=”auteur”>
       <xsl:value-of select=”concat(text(),' ')” />
     </xsl:for-each>
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Auteur : Chagnon
Auteur : Nielsen Chuang
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match=”/liste/livre”>
     Auteur :
     <xsl:for-each select=”auteur”>
       <xsl:value-of select=”text()” /><xsl:text> </xsl:text>
     </xsl:for-each>
  </xsl:template>

</xsl:stylesheet>
```

`<text> texte </text>`

- Prend du texte pur (sans < > &) et le produit en sortie, y compris les espaces.
- Si on ne l'utilise pas, rien n'est garanti pour les espaces et sauts de lignes.

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

webquantique

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/">
     <xsl:value-of select="/liste/livre[1]/@genre"/>
     <xsl:value-of select="/liste/livre[2]/@genre"/>
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

web quantique

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/">
     <xsl:value-of select="/liste/livre[1]/@genre"/>
     <xsl:text> </xsl:text>
     <xsl:value-of select="/liste/livre[2]/@genre"/>
  </xsl:template>

</xsl:stylesheet>
```

`<if test=”...”> exp </if>`

- L'attribut `test` prend une expression XPath de type booléen.
- L'expression `exp` est évaluée si `test` est vrai.

```
<choose>
  <when test="test1"> exp1 </when>
  <when test="test2"> exp2 </when>
  ...
  <otherwise> expn </otherwise>
</choose>
```

- Les attributs `test` prennent une expression XPath de type booléen.
- La seule expression `exp` à être évaluée correspond au premier `test` vrai.
- `expn` est évaluée si aucun des tests n'est vrai

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
web : Chouette sujet
quantique : Sujet bof
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/liste/livre">
     <xsl:value-of select='@genre' /><xsl:text> : </xsl:text>
     <xsl:if test="@genre='web'">Chouette sujet</xsl:if>
     <xsl:if test="@genre='quantique'">Sujet bof</xsl:if>
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
web : Chouette sujet
quantique : Sujet bof
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/liste/livre">
     <xsl:value-of select='@genre' /><xsl:text> : </xsl:text>
     <xsl:choose>
       <xsl:when test="@genre='web'">Chouette sujet</xsl:when>
       <xsl:when test="@genre='quantique'">Sujet bof</xsl:when>
       <xsl:otherwise>Rien à dire</xsl:otherwise>
     </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre='abc'>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
web : Chouette sujet
abc : Rien à dire
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/liste/livre">
     <xsl:value-of select='@genre' /><xsl:text> : </xsl:text>
     <xsl:choose>
       <xsl:when test="@genre='web'">Chouette sujet</xsl:when>
       <xsl:when test="@genre='quantique'">Sujet bof</xsl:when>
       <xsl:otherwise>Rien à dire</xsl:otherwise>
     </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

```
<element name="aaa"> exp </element>
<attribute name="aaa"> exp </attribute>
```

- Crée un élément (resp. un attribut) `aaa` contenant `exp`.
- L'attribut est ajouté à l'élément parent.
- Intéret : on peut créer un élément avec un nom non connu à l'avance
- Si `aaa` est une expression XPath, besoin de l'encadrer avec {...}.
- `namespace="uri"` peut être utilisé pour ajouter un espace de nom à l'élément.

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
<web>
  Blah
</web>
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/">
     <xsl:element name='{/liste/livre[1]/@genre}'>
        Blah
     </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
<web xmlns='http://mon-nom'>
  Blah
</web>
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/">
     <xsl:element name='{/liste/livre[1]/@genre}'
                  namespace='http://mon-nom'>
        Blah
     </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
<web lang=”fr”>
  Blah
</web>
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/">
     <xsl:element name='{/liste/livre[1]/@genre}'>
        <xsl:attribute name=”lang”>fr</xsl:attribute>
        Blah
     </xsl:element>
  </xsl:template>
</xsl:stylesheet>
```

Nul besoin de ça pour créer un élément connu

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
<biblio>
<volume>XML</volume>
<volume>Quantum Computation</volume>
</biblio>
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
  <xsl:template match="/">
     <biblio>
       <xsl:for-each select='/liste/livre'>
         <volume><xsl:value-of select=”titre” /></volume>
       </xsl:for-each>
     </biblio>
  </xsl:template>
</xsl:stylesheet>
```

Mais le document doit être valide quand même... **Attention aux espaces de noms**

<variable name='nom' select='exp' />

- Associe à nom le résultat de l'expression XPath exp.
- Réutilisable avec $name dans <value-of … />

```
<liste>
  <livre genre="web">
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre>
    <titre>Q. C.</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

```
<stylesheet version="1.0" xmlns="...">

 <template match="/liste/livre">
  <variable name='nom' select='titre'/>
  Titre : <value-of select='$nom' />
 </template>

</stylesheet>
```

Réponse :

```
Titre : XML
Titre : Quantum Computation
```

<copy-of select='exp' />

- Copie l'élément et tous les descendants résultat de l'expression XPath exp.

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre>
    <titre>Q. C.</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

```
<stylesheet version=”1.0” xmlns=”...”>

 <template match="/">
  <copy-of select='/liste/livre[1]'/>
 </template>

</stylesheet>
```

Réponse :

```
<livre genre=”web”>
  <titre>XML</titre>
  <auteur>Chagnon</auteur>
  <pub>2007</pub>
</livre>
```

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Turing</emprunt>
</lib>
```

Réponse :

XML

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
  <template match="/">
    <variable name='x' select='/lib/emprunt[.=”Valiron”]/@ref' />
    <value-of select='/lib/livre[@id=$x]/titre' />
  </template>
</stylesheet>
```

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

XML Quantum Computation

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'                Contient { id='a'  id='b' }
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

dans chaque cas, on se place sur le noeud contextuel

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

Contient `'a'` puis `'b'`

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

Affiche le titre correspondant : `$y = a` puis `b`

Exemple

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <variable name='x'
             select='/lib/emprunt[.="Valiron"]/@ref' />
   <for-each select='$x'>
     <variable name='y' select='.' />
     <value-of select='/lib/livre[@id=$y]/titre' />
     <text> </text>
   </for-each>
 </template>
</stylesheet>
```

Met un espace après chaque.

Exemple : sans $x

```
<lib>
  <livre id=”a”>
    <titre>XML</titre>
  </livre>
  <livre id=”b”>
    <titre>Quantum Computation</titre>
  </livre>
  <emprunt ref=”a”>Valiron</emprunt>
  <emprunt ref=”b”>Valiron</emprunt>
</lib>
```

Réponse :

```
XML Quantum Computation
```

doc.xsl :

```
<stylesheet version=”1.0”
          xmlns=”http://www.w3.org/1999/XSL/Transform”>
 <template match="/">
   <for-each select='/lib/emprunt[.="Valiron"]/@ref'> ← {id='a' id='b'}
     <variable name='y' select='.' />                 ← $y='a' puis $y='b'
     <value-of select='/lib/livre[@id=$y]/titre' /> ← ...
     <text> </text>                                   ← ...
   </for-each>
 </template>
</stylesheet>
```

<variable name='nom'> exp </variable>

- Associe à nom la valeur exp (pas d'évaluation).
  - Du texte
  - Des balises XML
- Réutilisable avec $name dans <value-of … /> dans <copy-of … /> (si contenu XML)

```
<stylesheet version="1.0" xmlns="...">

 <template match="/">
  <variable name='var'>Blah</variable>
  Je dis : <value-of select='$var' />
 </template>

</stylesheet>
```

Réponse :

```
Je dis : Blah
```

<apply-templates select='...' />

- Part du noeud contextuel et applique le modèle le plus adapté à l'ensemble de noeuds décrit par `select`.

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre>
    <titre>Q. C.</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

```
<stylesheet version=”1.0” xmlns=”...”>

  <template match="*">
    Un noeud <apply-templates />
  </template>

</stylesheet>
```

<apply-templates select='...' />

- Part du noeud contextuel et applique le modèle le plus adapté à l'ensemble de noeuds décrit par `select`.

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre>
    <titre>Q. C.</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

```
<stylesheet version=”1.0” xmlns=”...”>

  <template match="*">
    Un noeud : <apply-templates
                    select='*|text()'/>
  </template>

  <template match="text()">
    <value-of select='.' />
  </template>

</stylesheet>
```

<apply-templates />

- Dans ce cas, `select` est par défaut `*|text()`

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre>
    <titre>Q. C.</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

```
<stylesheet version=”1.0” xmlns=”...”>

  <template match="*">
    Un noeud : <apply-templates />
  </template>

  <template match="text()">
    <value-of select='.' />
  </template>

</stylesheet>
```

Modèle par défaut

```
<stylesheet version=”1.0” xmlns=”...”>

</stylesheet>
```

=

```
<stylesheet version=”1.0” xmlns=”...”>

  <template match="/|*">
    <apply-templates />
  </template>

  <template match="text()|@*">
    <value-of select='.' />
  </template>

</stylesheet>
```

Revoyons : par défaut

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
XML
Chagnon
2007


Quantum Computation
Nielsen
Chuang
2001
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

</xsl:stylesheet>
```

Revoyons : par défaut

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
XML
Chagnon
2007


Quantum Computation
Nielsen
Chuang
2001
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/|*">
    <xsl:apply-templates />
  </xsl:template>

  <xsl:template match="text()|@*">
    <xsl:value-of select='.' />
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

On réécrit...

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/liste/livre[@genre='web']">
    Trouvé
  </xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Trouvé


Quantum Computation
Nielsen
Chuang
2001
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/liste/livre[@genre='web']">
    Trouvé
  </xsl:template>

  <xsl:template match="/|*"><xsl:apply-templates /></xsl:template>

  <xsl:template match="text()|@*">
       <xsl:value-of select='.' /></xsl:template>

</xsl:stylesheet>
```

Exemple

```
<liste>
  <livre genre=”web”>
    <titre>XML</titre>
    <auteur>Chagnon</auteur>
    <pub>2007</pub>
  </livre>
  <livre genre=”quantique”>
    <titre>Quantum Computation</titre>
    <auteur>Nielsen</auteur>
    <auteur>Chuang</auteur>
    <pub>2001</pub>
  </livre>
</liste>
```

Réponse :

```
Trouvé
```

doc.xsl :

```
<xsl:stylesheet version=”1.0”
          xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>

  <xsl:template match="/liste/livre[@genre='web']">
    Trouvé
  </xsl:template>

  <template match="text()|@*"></template>

</xsl:stylesheet>
```