

Modélisation et interopérabilité :

Semaine 46, cours 8

Benoît Valiron <benoit.valiron@monoidal.net>

<http://inf356.monoidal.net/>

1

XPath

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

2

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`

3

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`

4

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']`

5

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']`

6

XSLT

XSLT : Transformations XML

- 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`

7

8

```
<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.

9

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.

11

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>

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>
```

Réponse :
Livre trouvé
Livre trouvé

10

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>

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>
```

Réponse :
Réponse 1
Réponse 1

12

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>
```

13

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...](#)

14

`<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>
```

15

16

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 :
Livres web
Livres quantique

doc.xsl :

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

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

</xsl:stylesheet>
```

17

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 :
Livres "XML
Chagnon
2007"

doc.xsl :

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

  <xsl:template match="/liste">
    Livres "<xsl:value-of select="livre" />"
  </xsl:template>

</xsl:stylesheet>
```

18

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>
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>
```

Réponse :

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

19

`<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.

20

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>
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>
```

Réponse :

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

21

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>
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>
```

Réponse :

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

22

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>
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>
```

Réponse :

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

23

`<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.

24

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>
```

25

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>
```

26

```
<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

27

28

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>
```

29

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>
```

30

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>

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>
```

Réponse :

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

31

```
<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.
- Intérêt : 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.

32

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>

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>
```

Réponse :

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

33

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>

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>
```

Réponse :

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

34

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>

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>
```

Réponse :

```
<web lang="fr">
  Blah
</web>
```

35

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>

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>
```

Réponse :

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

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

36

<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
```

37

<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>
```

38

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>
```

39

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' />
    </for-each>
  </template>
</stylesheet>
```

40

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' />
    </for-each>
  </template>
</stylesheet>
```

41

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' />
    </for-each>
  </template>
</stylesheet>
```

42

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='$x'>
      <variable name='y' select='.' /> Contient 'a' puis 'b'
      <value-of select='/lib/livre[@id=$y]/titre' />
      <text> </text>
    </for-each>
  </template>
</stylesheet>
```

43

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> Affiche le titre correspondant : $y = a puis b
    </for-each>
  </template>
</stylesheet>
```

44

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> Met un espace après chaque.
    </for-each>
  </template>
</stylesheet>
```

45

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' >
      <variable name='y' select='.' />
      <value-of select='/lib/livre[@id=$y]/titre' />
      <text> </text>
    </for-each>
  </template>
</stylesheet>
```

46

<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

47

<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>
```

48

<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>
```

49

<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>
```

50

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>
```

51

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>
```

52

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>
```

53

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>
```

54

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>
```

55

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>
```

56