

XSLT

Laurent Réveillère

Enseirb
Département Télécommunications

`Laurent.Reveillere@enseirb.fr`
`http://www.enseirb.fr/~reveille`

Affichage de document XML

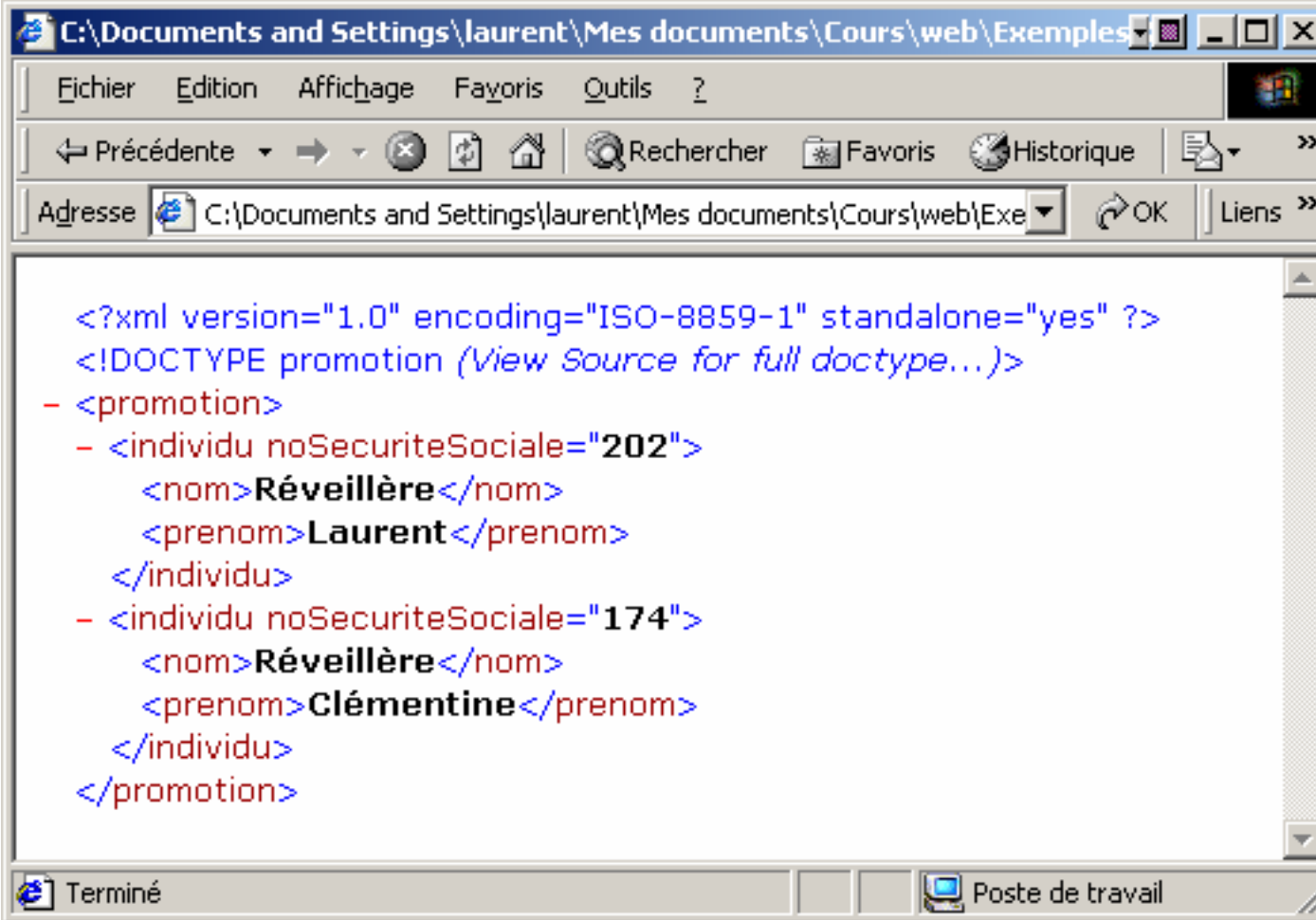
promotion.dtd

```
<!ELEMENT promotion (individu)+ >
<!ELEMENT individu ( nom , prenom ) >
<!ELEMENT nom (#PCDATA) >
<!ELEMENT prenom (#PCDATA) >
<!ATTLIST individu noSecuriteSociale ID #REQUIRED >
```

promotion.xml

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<!DOCTYPE promotion SYSTEM "promotion.dtd" >
<promotion>
  <individu noSecuriteSociale="202">
    <nom>Réveillère</nom>
    <prenom>Laurent</prenom>
  </individu>
  <individu noSecuriteSociale="174">
    <nom>Réveillère</nom>
    <prenom>Clémentine</prenom>
  </individu>
</promotion>
```

Résultat (IE)



The screenshot shows an Internet Explorer window with the address bar containing the path: C:\Documents and Settings\laurent\Mes documents\Cours\web\Exemples. The main content area displays the following XML code:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<!DOCTYPE promotion (View Source for full doctype...)>
- <promotion>
- <individu noSecuriteSociale="202">
  <nom>Réveillère</nom>
  <prenom>Laurent</prenom>
</individu>
- <individu noSecuriteSociale="174">
  <nom>Réveillère</nom>
  <prenom>Clémentine</prenom>
</individu>
</promotion>
```

The status bar at the bottom indicates "Terminé" and "Poste de travail".

Feuilles de style

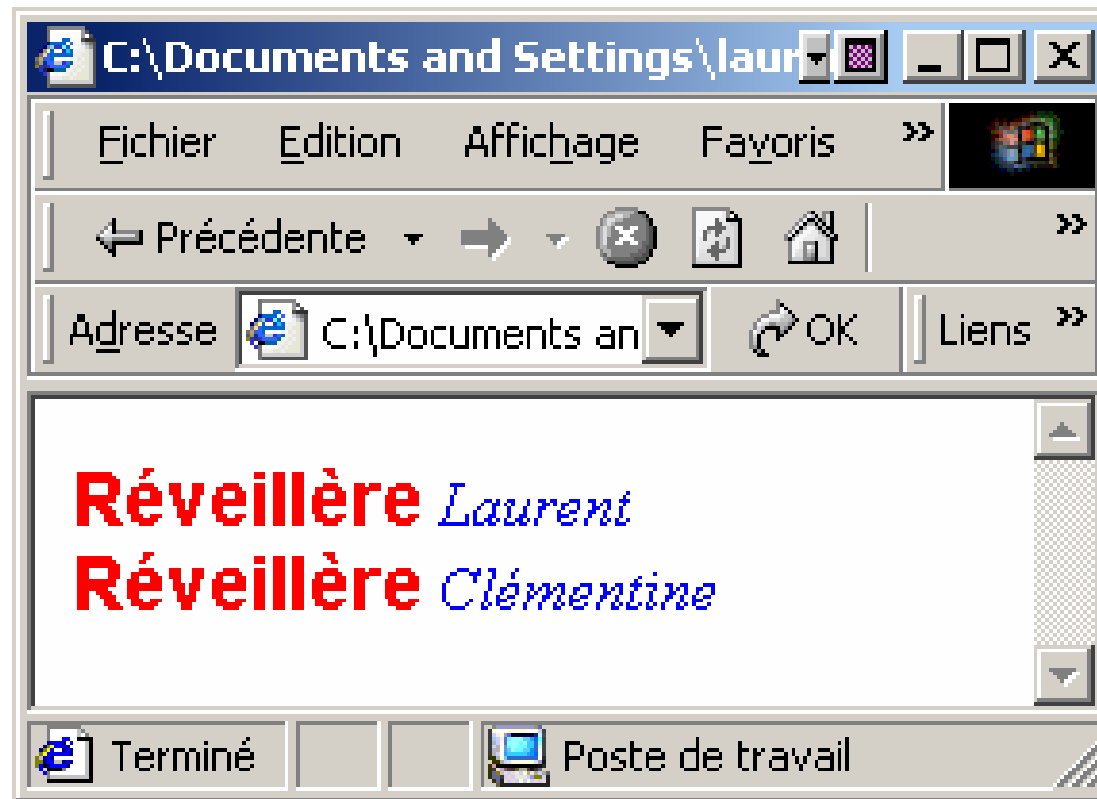
promotion.css

```
individu { display: block; }  
nom      { font-family: Arial; font-size: 14pt;  
          font-weight: bold; color: red; }  
prenom   { font-style: italic; color: blue; }
```

promotion2.xml

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>  
<?xml-stylesheet href="promotion.css" type="text/css" ?>  
<!DOCTYPE promotion SYSTEM "promotion.dtd" >  
<promotion>  
  [...]  
</promotion>
```

Résultat (IE)



→ limitation par les capacités de CSS

XSL : eXtensible Stylesheet Language

- ◆ Langage de définition du rendu graphique d'un document XML
 - XSLT (Transformation)
 - » langage de transformation d'un document XML
 - XSL FO (Formatting Objects)
 - » directives de mises en forme (\equiv CSS)
 - XSLT et XSL FO sont exprimés en XML
- ◆ Nouvelles solutions pour afficher un doc. XML
 - XSLT → transformation XML en XHTML
 - XSL FO → interprétation XML
 - XSLT + XSL FO → transf. XML + interprétation

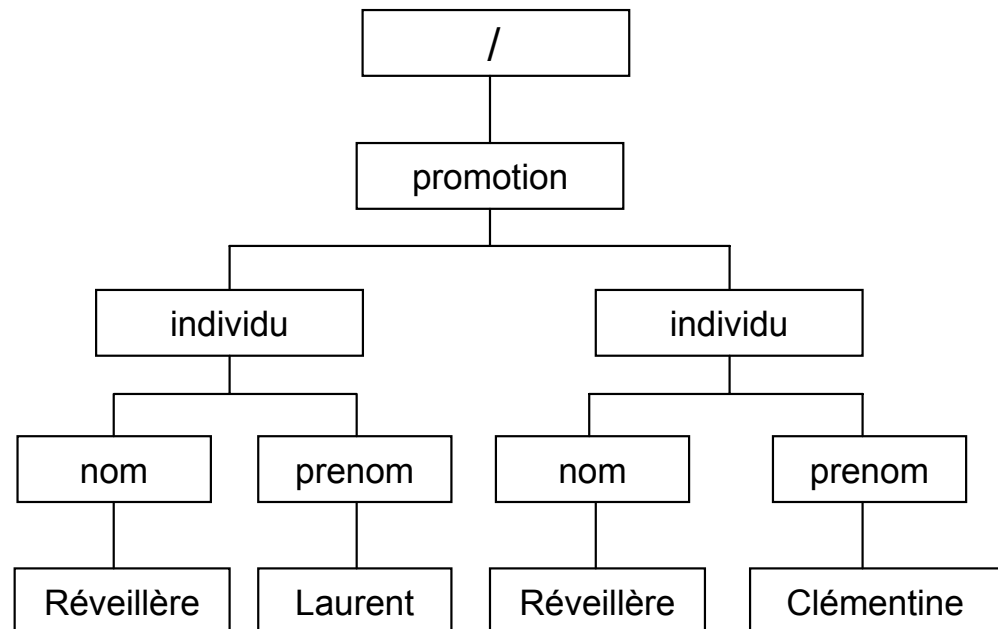
Principe XSLT

promotion.xml

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<promotion>
  <individu> <nom>Réveillère</nom> <prenom>Laurent</prenom> </individu>
  <individu> <nom>Réveillère</nom> <prenom>Clémentine</prenom> </individu>
</promotion>
```

- racine : *fictive*
- Noeud : balise incluse
- Feuille : contenu

Notion d'ancêtres
Notion de descendants



Règles

- ◆ *Pattern matching* sur un document XML
- ◆ règle \equiv balise + actions

```
<xsl:template match="une balise à détecter">  
    action à entreprendre lorsque la balise est rencontrée  
</xsl:template>
```

- La balise est une expression XPATH (voir plus loin)

- ◆ **Algorithme**

- pour toutes les règles

- pour les toutes les balises identiques à celle de la règle

- appliquer les actions de la règle

- sur le **sous-arbre de la balise**

Actions

- ◆ Balises à générer
- ◆ Commandes XSLT
- ◆ Principales commandes XSLT

```
<xsl:value-of select="expression de chemin" />
```

- sélectionne le noeud correspondant à l'expression de chemin XPath dans le sous-arbre de la balise

```
<xsl:apply-templates />
```

- réexamine les règles avec le sous-arbre de la balise

Exemple

promotion.xls

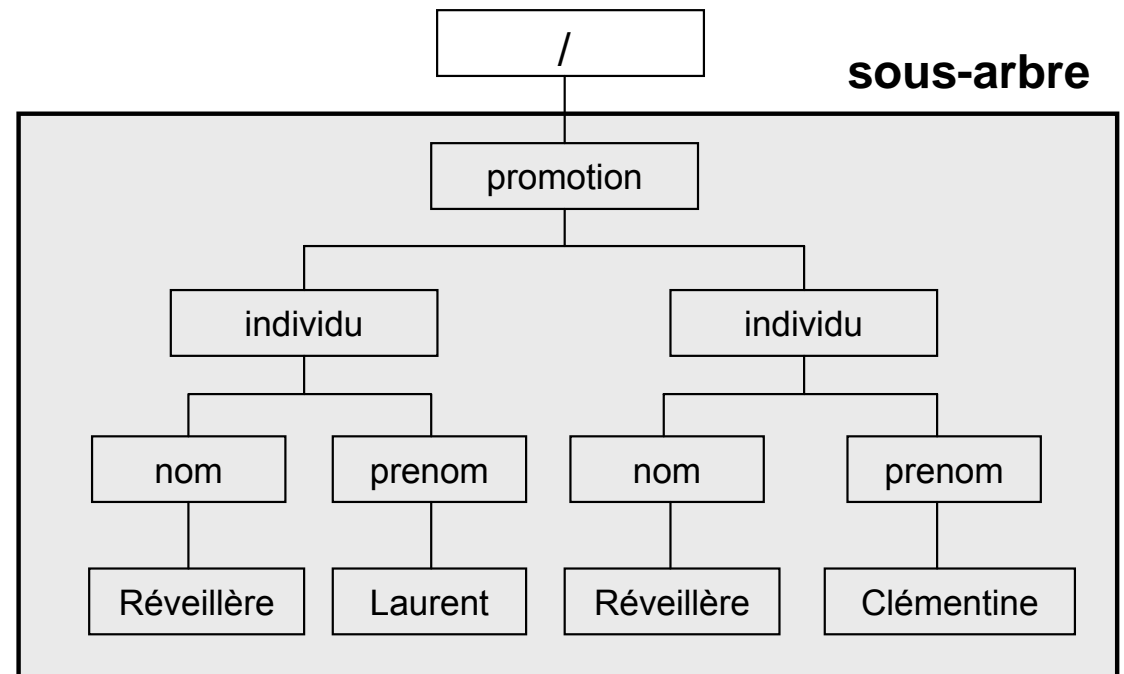
```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  <xsl:template match="/">
    <h1>Le 1er nom est
      <xsl:value-of select="promotion/individu/nom" />
    </h1>
  </xsl:template>
</xsl:stylesheet>
```

promotion3.xml

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<?xml-stylesheet href="promotion.xls" type="text/xsl" ?>
<promotion>
  [...]
</promotion>
```

Principe

- ◆ Application de la première règle sur la racine
- ◆ Génération de
`<h1>Le 1er nom est`
- ◆ Génération du noeud correspondant au chemin `promotion/individu/nom`
- ◆ Génération de
`</h1>`

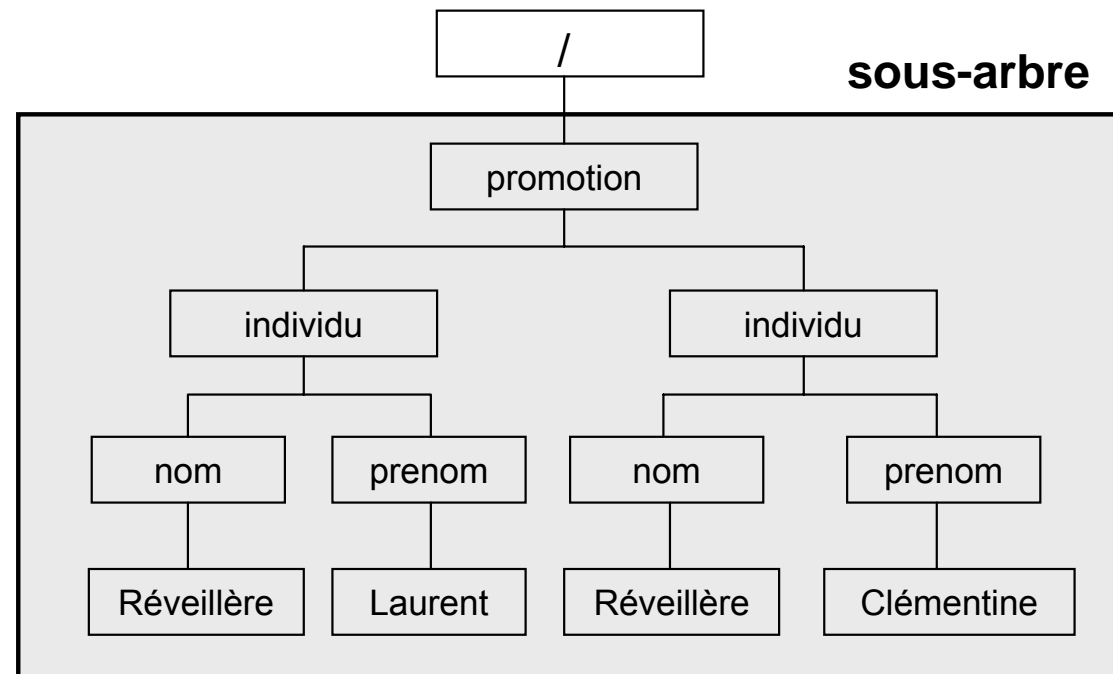
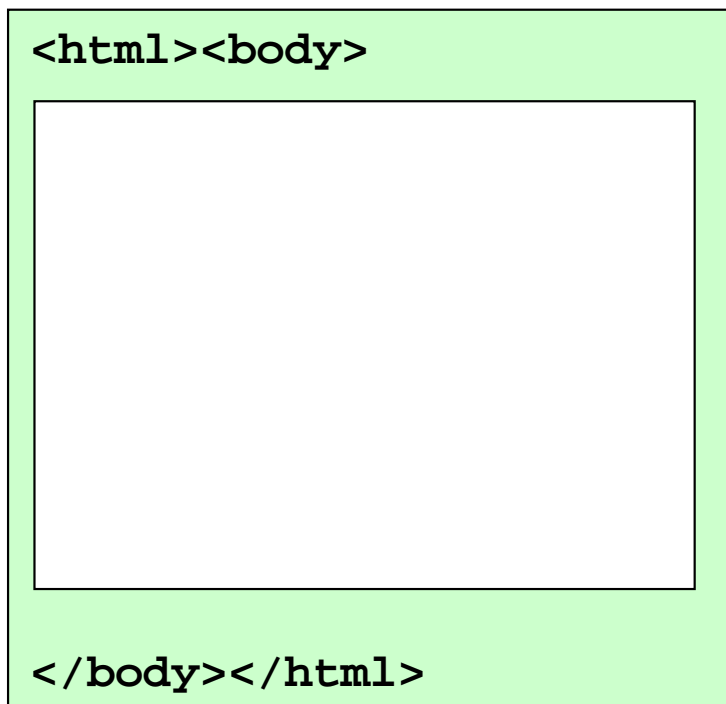


Exemple

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
  <xsl:template match="/">
    <html><body> <xsl:apply-templates/> </body></html>
  </xsl:template>
  <xsl:template match="promotion">
    <table border="1"> <xsl:apply-templates/> </table>
  </xsl:template>
  <xsl:template match="individu">
    <tr> <xsl:apply-templates/> </tr>
  </xsl:template>
  <xsl:template match="nom">
    <td> <xsl:value-of select="."/> </td>
  </xsl:template>
  <xsl:template match="prenom">
    <td> <xsl:value-of select="."/> </td>
  </xsl:template>
</xsl:stylesheet>
```

Étape 1

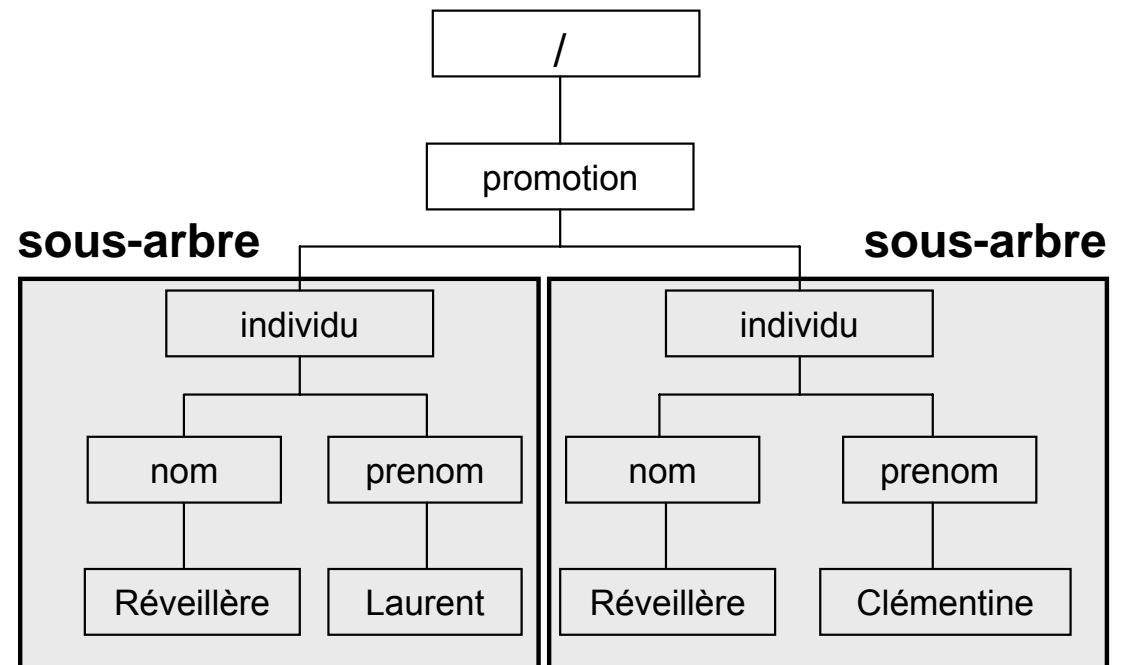
```
<xsl:template match="/">
  <html><body> <xsl:apply-templates/> </body></html>
</xsl:template>
```



Étape 2

```
<xsl:template match="promotion">
  <table border="1"> <xsl:apply-templates/> </table>
</xsl:template>
```

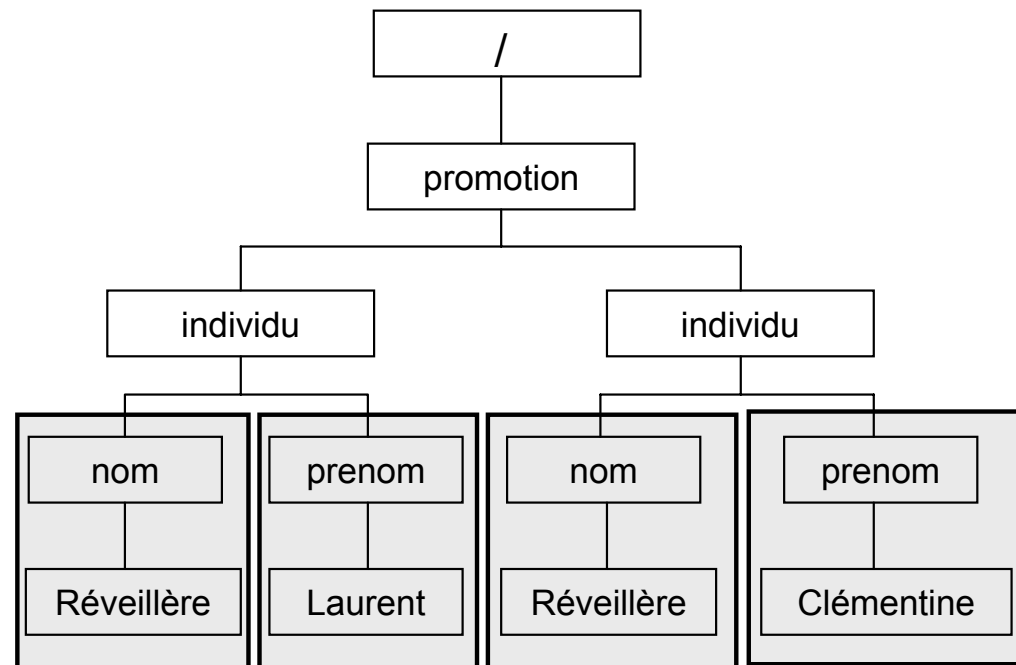
```
<html><body>
<table border="1">
</table>
</body></html>
```



Étape 3

```
<xsl:template match="individu">
  <tr> <xsl:apply-templates/> </tr>
</xsl:template>
```

```
<html><body>
<table border="1">
  <tr>
    <td>
  </td>
</tr>
<tr>
    <td>
  </td>
</tr>
</table>
</body></html>
```

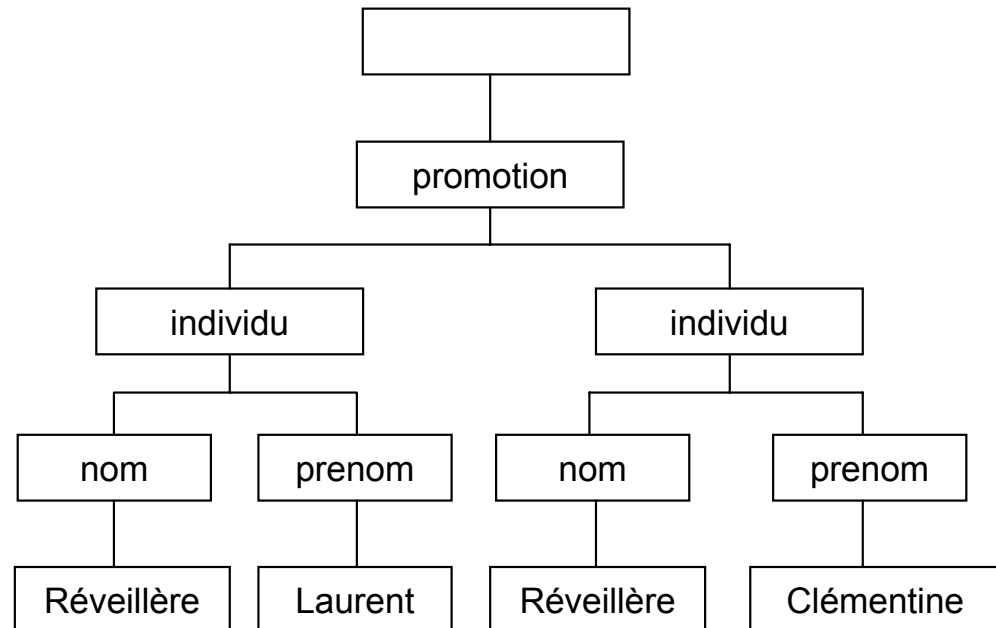


Étape 4

```
<xsl:template match="nom">
  <td> <xsl:value-of select="."/>
</td>
</xsl:template>
```

```
<xsl:template match="prenom">
  <td> <xsl:value-of select="."/>
</td>
</xsl:template>
```

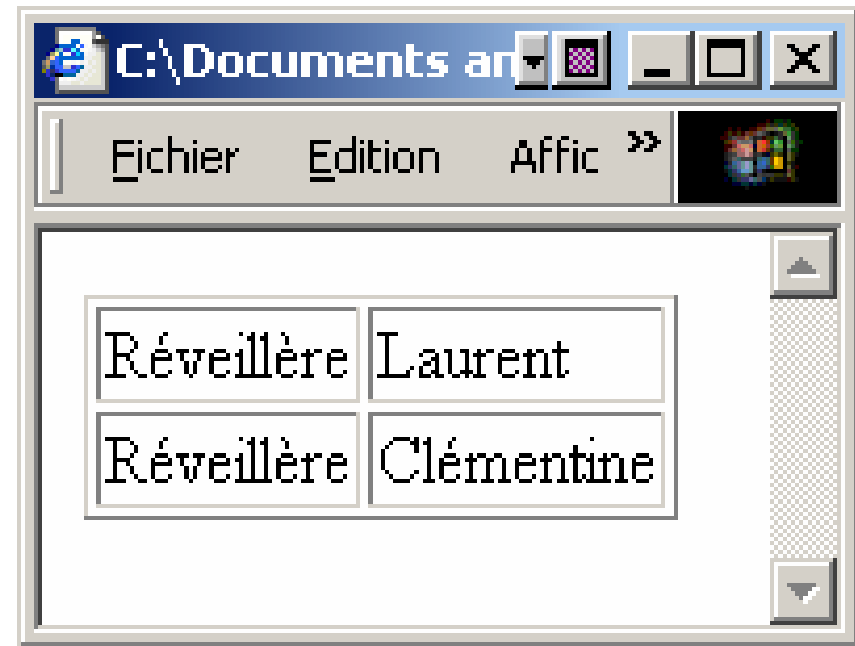
```
<html><body>
<table border="1">
  <tr>
    <td>Réveillère</td>
    <td>Laurent</td>
  </tr>
  <tr>
    <td>Réveillère</td>
    <td>Clémentine</td>
  </tr>
</table>
</body></html>
```



Résultat

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<promotion>
  <individu> <nom>Réveillère</nom> <prenom>Laurent</prenom> </individu>
  <individu> <nom>Réveillère</nom> <prenom>Clémentine</prenom> </individu>
</promotion>
```

```
<html><body>
<table border="1">
  <tr>
    <td>Réveillère</td>
    <td>Laurent</td>
  </tr>
  <tr>
    <td>Réveillère</td>
    <td>Clémentine</td>
  </tr>
</table>
</body></html>
```



Autres commandes

- ◆ `<xsl:for-each select="pattern">`
 - sélectionne l'ens. des parties du doc. correspondant au *pattern*
- ◆ `<xsl:process>`
 - à l'intérieur d'un `<xsl:for-each>` fournit l'élément suivant dans l'ensemble des parties du doc.
 - ➔ permet d'itérer sur l'ensemble
- ◆ `<xsl:if test="pattern"> ... </xsl:if>`
 - test si un *pattern* est présent ou non
- ◆ `<xsl:counter name="string">`
 - définit un compteur
- ◆ `<xsl:counter-increment name="string">`
 - incrémente un compteur
- ◆ `<xsl:counter-reset name="string">`
 - remet à zéro un compteur

XPath

Laurent Réveillère

Enseirb
Département Télécommunications

`Laurent.Reveillere@enseirb.fr`
`http://www.enseirb.fr/~reveille`

Introduction

- ◆ XPATH : langage permettant d'identifier un élément dans un arbre XML
 - même principe que la désignation d'un chemin dans une arborescence de fichier
- ◆ Motifs de base
 - / **nœud racine**
 - . **nœud courant**
 - .. **nœud parent**
 - /promotion **nœud** promotion **sous le nœud racine**
 - ./nom **nœud** nom **sous le nœud courant**

Principaux motifs

`individu/nom`

`promotion//nom`

`promotion//nom|prenom`

`promotion/*`

`*/promotion`

`individu[nom]/prenom`

`individu[@noss]`

`individu[@noss="203"]`

nœud nom **fils de** individu

nœud nom **ayant le nœud**

promotion **pour aïeul**

nœud nom **ou** prenom **ayant le**

nœud promotion **pour aïeul**

nœud fils du **nœud** promotion

nœud père du **nœud** promotion

nœud prenom **fils du nœud**

individu **ayant un fils** nom

nœud individu **ayant un attribut**

noss

nœud individu **ayant un attribut**

noss **valant** 203