

TP WEB DES DONNEES N°4

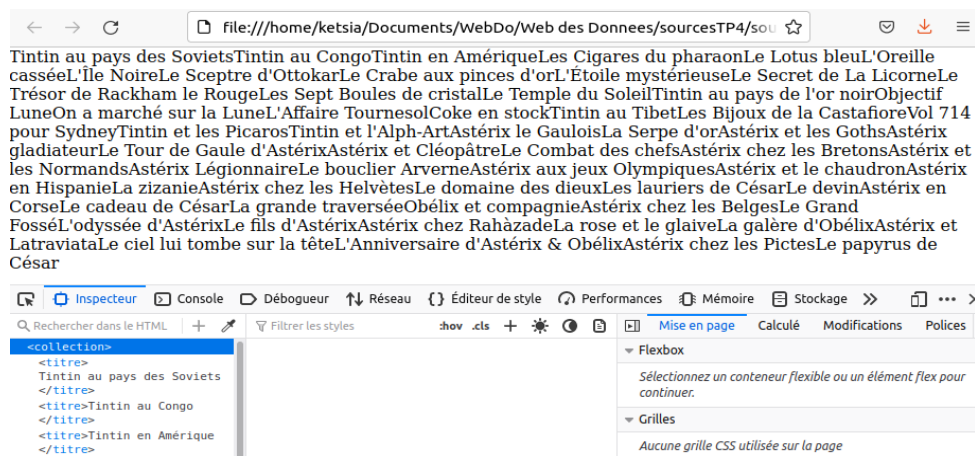
Mulapi Tita Ketsia

1.1 Exemple 1

```
<xsl:template match="/">
<collection>
<xsl:copy-of select="//titre"/>
</collection>
</xsl:template>
```

ce patterne renvoie une collection (un ensemble) de titres où qu'ils soient dans notre fichier album.xml, sans les séparer (vu dans le navigateur, mais bien évidemment, techniquement via la programmation, on peut par exemple à l'aide d'une api SOAP, récupérer ces données d'une façon propre ou même avec du HTML les afficher de façon à expliciter le fait qu'ils ne sont en réalité pas concaténés).

<collection><titre></titre></collection>

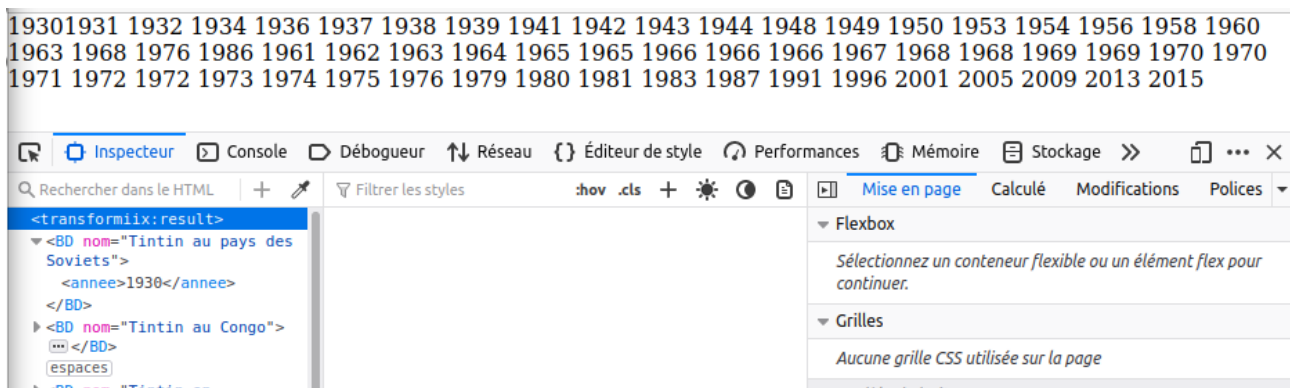


1.2 Exemple 2

```
<xsl:template match="//album">
  <BD>
    <xsl:attribute name="nom">
      <xsl:value-of select="titre"/>
    </xsl:attribute>
    <xsl:copy-of select="date/annee"/>
  </BD>
</xsl:template>
```

On obtient un nœud « **BD** » avec un attribut « **nom** » associé à une valeur « **titre** », le tout contenant une « **date/annee** » à partir de « **album** » présent où que ce soit dans « **albums** » :

<BD nom=titre> <annee>date</annee> </BD>



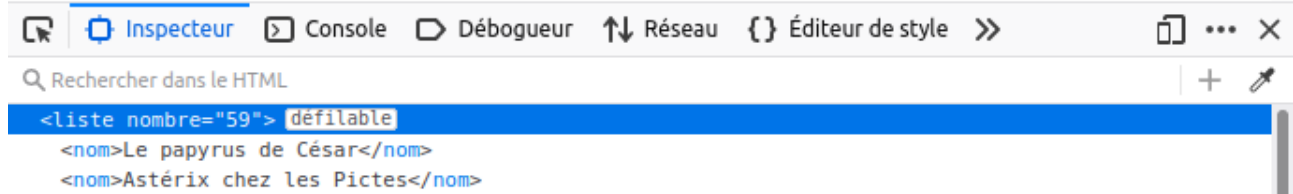
1.3 Exemple 3

xsl : sort permet trier un ensemble de valeurs.

```
<xsl:template match="/">
  <liste>
    <xsl:attribute name="nombre">
      <xsl:value-of select="count(albums/album)"/>
    </xsl:attribute>
    <xsl:for-each select="albums/album">
      <xsl:sort select="date/annee" order="descending"/>
      <xsl:variable name="album" select="current()"/>
      <nom>
        <xsl:value-of select="$album/titre"/>
      </nom>
    </xsl:for-each>
  </liste>
</xsl:template>
```

On veut obtenir le « **titre** » de chaque album selon l'ordre décroissant des « **années** », ces titres, contenu dans l'élément « **nom** », sont itérativement parcouru dans un élément xsl nommé « **liste** » qui représente l'ensemble des sous nœuds de albums et, qui a pour attribut le nombre (total) d'album présent dans le fichier albums.xml

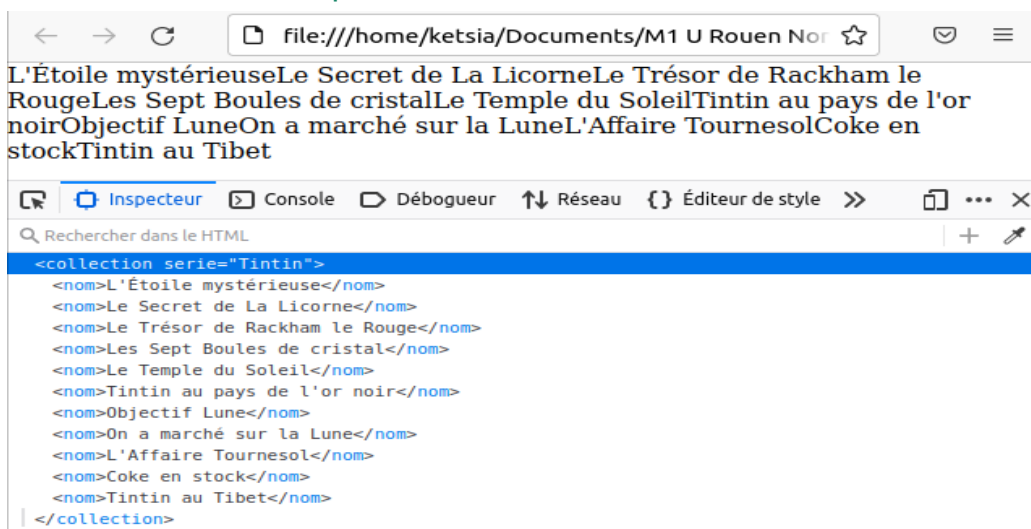
Le papyrus de CésarAstérix chez les PictesL'Anniversaire d'Astérix & ObélixLe ciel lui tombe sur la têteAstérix et LatraviataLa galère d'ObélixLa rose et le glaiveAstérix chez RahazadeTintin et l'Alph-ArtLe fils d'AstérixL'odyssée d'AstérixLe Grand FosséAstérix chez les BelgesTintin et les PicarosObélix et compagnieLa grande traverséeLe cadeau de CésarAstérix en CorseLes lauriers de CésarLe devinLe domaine des dieuxLa zizanieAstérix chez les HelvètesAstérix et le chaudronAstérix en HispanieVol 714 pour SydneyLe bouclier ArverneAstérix aux jeux OlympiquesAstérix LégionnaireLe Combat des chefsAstérix chez les



1.4 Exemple 4

```
<xsl:template match="/">
  <collection serie="Tintin">
    <xsl:apply-templates
      select="/albums/album[@serie='Tintin' and @numero>=10 and @numero<=20]"/>
    </collection>
  </xsl:template>
  <xsl:template match="album">
    <nom>
      <xsl:value-of select="titre"/>
    </nom>
  </xsl:template>
```

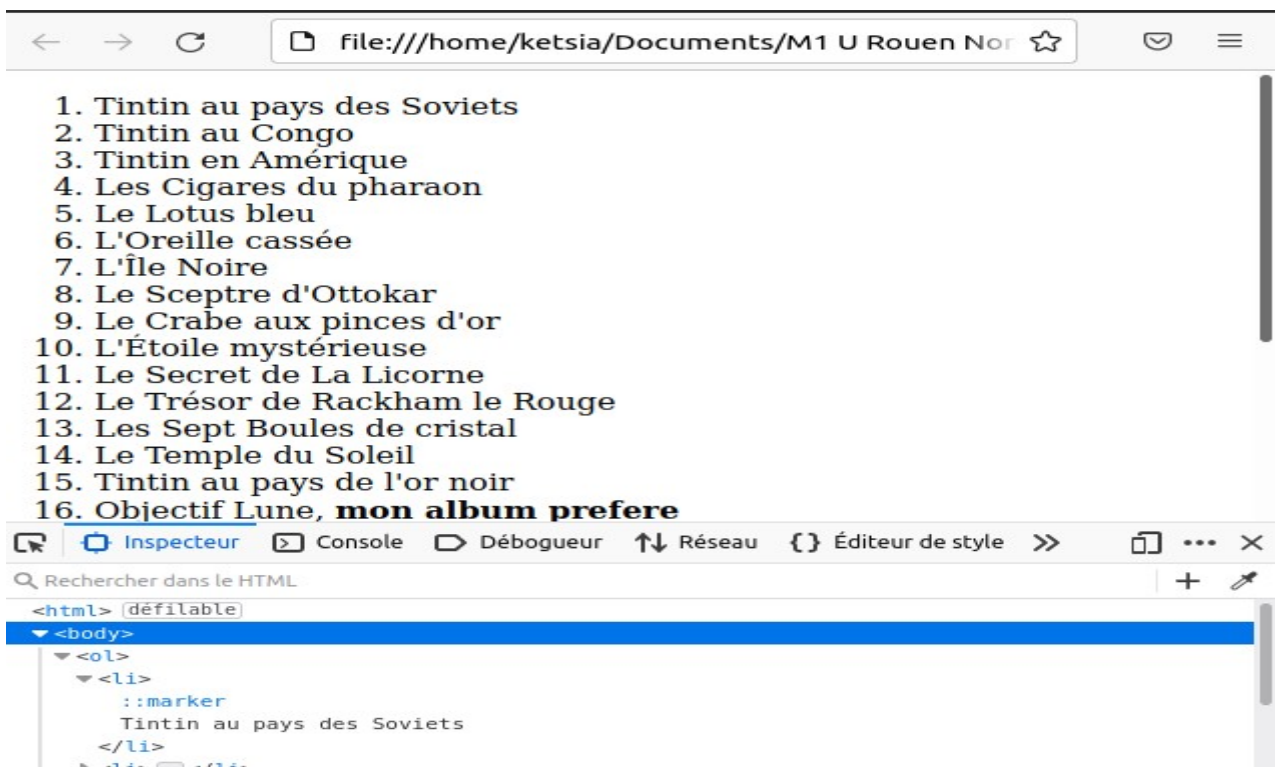
On récupère l'ensemble d'« **album** » dont « **la série est tintin** » et, l'attribut « **numéro** » a une valeur comprise « **entre 10 et 20** » et, on se contente de récupérer le « **titre** » de chacun de ces albums, que l'on met par la suite dans un élément « **nom** » pour enfin les afficher.



1.5 Exemple 5

```
<xsl:output method="html"/>
<xsl:template match="/">
<html>
<body>
<ol>
<xsl:apply-templates select="/albums/album[@serie='Tintin']"/>
</ol>
</body>
</html>
</xsl:template>
<xsl:template match="album">
<li>
<xsl:value-of select="titre"/>
<xsl:if test="@numero=16">
<b>mon album prefere</b>
</xsl:if>
</li>
</xsl:template>
```

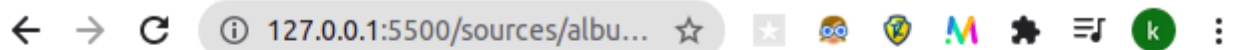
On récupère les « **album** » dont la « **série est tintin** », on place chaque album dans une « **liste (li) numéroté dans l'ordre croissant (ol)** », en y mettant le « **titre** » de l'album associé.



1.6 Exemple 6

```
<xsl:output method="text"/>
<xsl:template match="/">
Voici ma collection de Tintin :
<xsl:apply-templates select="/albums/album[@serie='Tintin']"/>
</xsl:template>
<xsl:template match="album">
<xsl:value-of select="titre"/>
<xsl:choose>
<xsl:when test="position()=last()">
<xsl:text>.</xsl:text>
</xsl:when>
<xsl:otherwise>
<xsl:text>,</xsl:text>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
```

On récupère les « **titres** » des albums dont la « **série est tintin** » qu'on sépare par une virgule tant qu'on est pas arrivé à la fin, sinon on place un point. Et on obtient en sortie du texte.



Voici ma collection de Tintin :

Tintin au pays des Soviets, Tintin au Congo, Tintin en Amérique, Les Cigi

1.7 Exemple 7

xsl:key (**tintin**) permet de mémoriser un index (**élément**) faisant le lien entre des éléments avec des valeurs communes. Bref, un index est donc un ensemble de couple valeur-clé (valeur, liste des éléments ayant cette valeur qui sont des clés).

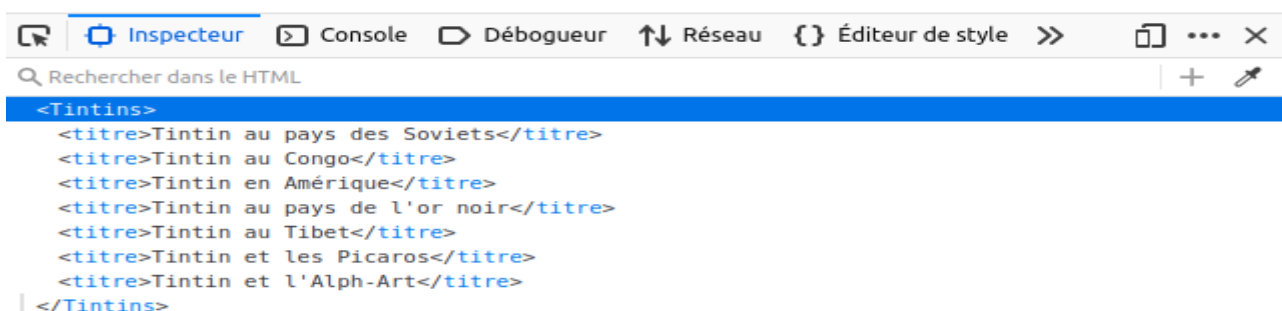
```
<xsl:output method="xml"/>
<!--On a des albums[@serie=tintin] tel que
(index debut titre=<album>) avec comme clé=tintin (1,6)-->
<xsl:key name="index debut titre"
match="//album[@serie='Tintin']" use="substring(titre,1,6)"/>
<xsl:template match="/">
<Tintins>
<!--Pour chaque album qui a aussi la clé tintin → récupérer le et,-->
<xsl:for-each select="key('index debut titre', 'Tintin')">
<!-- afficher tous les élément album/titre -- >
<xsl:copy-of select="titre"/>
</xsl:for-each>
</Tintins>
</xsl:template>
```

On mémorise un index qui s'appelle « **index_debut_titre** » et qui contient un ensemble d'album associé à l'extraction d'une sous chaîne qui débute à la position 1 et, qui est de longueur 6 de chaque titre et qui n'est rien d'autre que « tintin ».

Index (Élément) : index_debut_titre (concrètement ici ce sont des titres)	clé
Tintin au pays des Soviets	Tintin
...	...
Tintin et l'Alph-Art	Tintin

Bref : C'est donc « **la liste de album qui devient Tintins** » de « **série Tintin** », qui sont des « **éléments titre** », associé à « **la clé tintin** » qui est une sous chaîne de ces titres.

Tintin au pays des SovietsTintin au CongoTintin en AmériqueTintin au pays de l'or noirTintin au TibetTintin et les PicarosTintin et l'Alph-Art



1.8 Exemple 8

```
<xsl:key name="groupes" match="//album"
use="date/annee/text()" />
<xsl:template match="/">
<calendrier>
<xsl:for-each select="//album[
generate-id()=generate-id(key('groupes', date/annee/text())[1])]">
<xsl:sort select="date/annee" order="descending"/>
<annee>
<xsl:value-of select="date/annee"/>
<xsl:text> : </xsl:text>
<xsl:for-each select="key('groupes', date/annee/text())">
<xsl:text>album </xsl:text>
<xsl:value-of select="@numero"/>
<xsl:text>, </xsl:text>
</xsl:for-each>
</annee>
</xsl:for-each>
</calendrier>
</xsl:template>
```

* Analyse technique :

- 1) Ici, on se base sur le fait que chaque « **album** » possède un identifiant interne (caché) qui est le « **numéro** » et qui le distingue des autres. « **generate-id()** » nous permet d'obtenir cet identifiant" pour chaque album courant.
- 2) Et donc, en faisant « **album[generate-id()=generate-id(key('elt-album-group',annee-txt)[1])]** » on obtient l'id du 1^{er} élément indexé par l'album courant, que l'on compare à l'id de l'album passé en paramètre.
- 3) « **On ordonne de façon décroissante** » les « **éléments-années** » que l'on affichera ensuite et qui sera suivit de « **:** »
- 4) On parcourt cet élément « **année** » qui doit correspondre à la « **clé-année** », afin d'afficher chaque « **album et son numéro associé** », tant que ceux-ci sont sortis à la même année, ils seront alors séparés par « **une virgule** ».

* Résultat

On obtient « **un calendrier** » qui contient un ensemble « **d'élément-année** » tel que ceux-ci contient « **le.s numéro.s** » correspondant au.x album.s sortis à cette même année.

2015 : album 36, 2013 : album 35, 2009 : album 34, 2005 : album 33, 2001 : album 31, 1996 : album 30, 1991 : album 29, 1987 : album 28, 1986 : album 24, 1983 : album 27, 1981 : album 26, 1980 : album 25, 1979 : album 24, 1976 : album 23, album 23, 1975 : album 22, 1974 : album 21, 1973 : album 20, 1972 : album 18, album 19, 1971 : album 17, 1970 : album 15, album 16, 1969 : album 13, album 14, 1968 : album 22, album 11, album 12, 1967 : album 10, 1966 : album 7, album 8, album 9, 1965 : album 5, album 6, 1964 :

Inspecteur Console Débugueur Réseau Éditeur de style

Rechercher dans le HTML

```
<calendrier> [défilable]
  <annee>2015 : album 36,</annee>
  <annee>2013 : album 35,</annee>
  <annee>2009 : album 34,</annee>
  <annee>2005 : album 33,</annee>
  <annee>2001 : album 31,</annee>
  <annee>1996 : album 30,</annee>
  <annee>1991 : album 29,</annee>
  <annee>1987 : album 28,</annee>
  <annee>1986 : album 24,</annee>
  <annee>1983 : album 27,</annee>
  <annee>1981 : album 26,</annee>
  <annee>1980 : album 25,</annee>
  <annee>1979 : album 24,</annee>
  <annee>1976 : album 23, album 23,</annee>
  <annee>1975 : album 22,</annee>
  <annee>1974 : album 21,</annee>
  <annee>1973 : album 20,</annee>
  <annee>1972 : album 18, album 19,</annee>
  <annee>1971 : album 17,</annee>
  <annee>1970 : album 15, album 16,</annee>
  <annee>1969 : album 13, album 14,</annee>
  <annee>1968 : album 22, album 11, album 12,</annee>
  <annee>1967 : album 10,</annee> [débordement]
  <annee>1966 : album 7, album 8, album 9,</annee> [débordement]
  <annee>1965 : album 5, album 6,</annee> [débordement]
```

calendrier > annee

2 Vos propres XSLT

2.1 Exercice 1

On veut obtenir quelque chose comme ça :

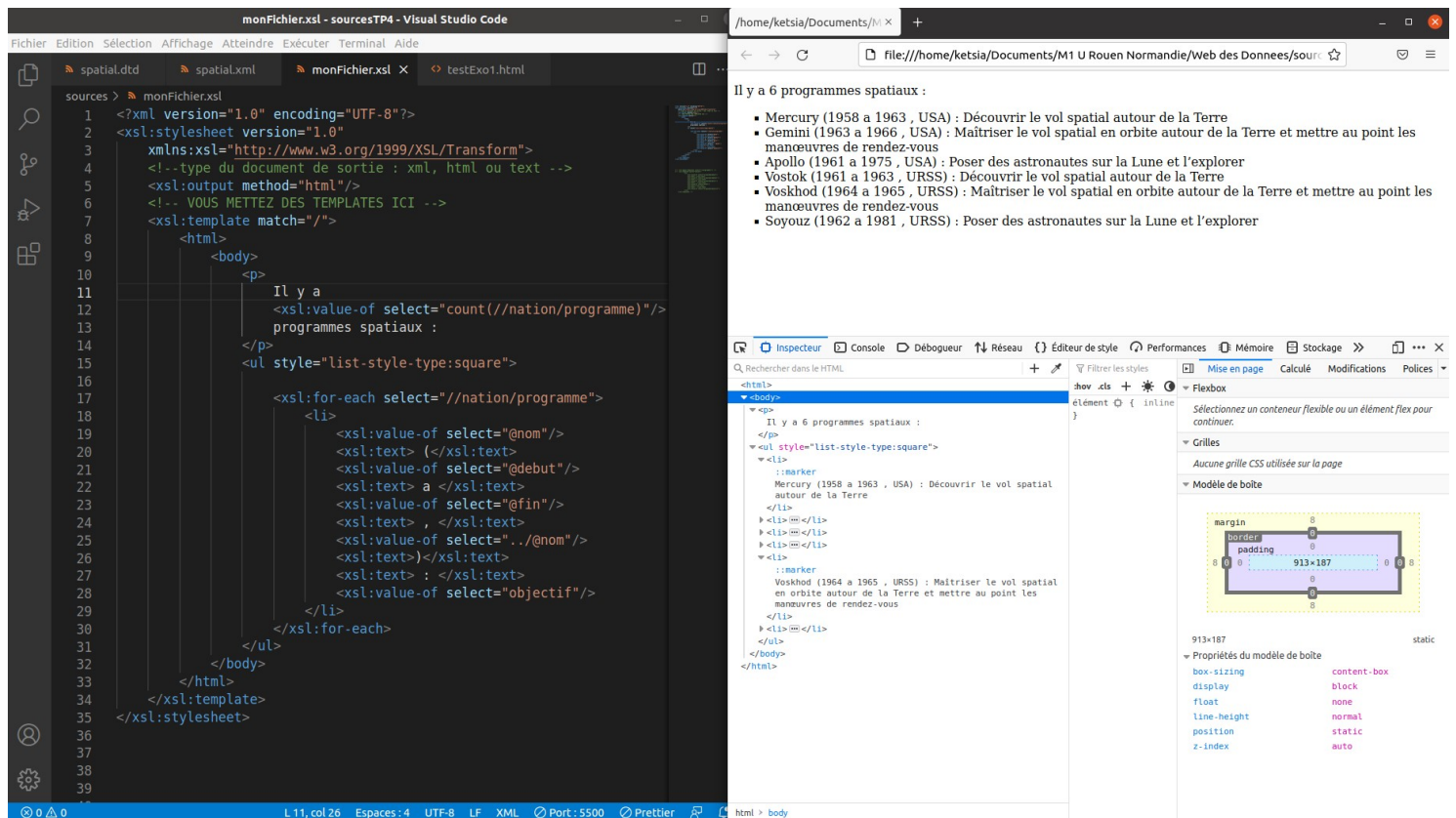
Il y a 6 programmes spatiaux :

- Mercury (1958 a 1963 , USA) : Decouvrir l e vol s p a t i a l autour de la Terre
- Gemini (1963 a 1966 , USA) : Maitriser l e vol s p a t i a l en orbite

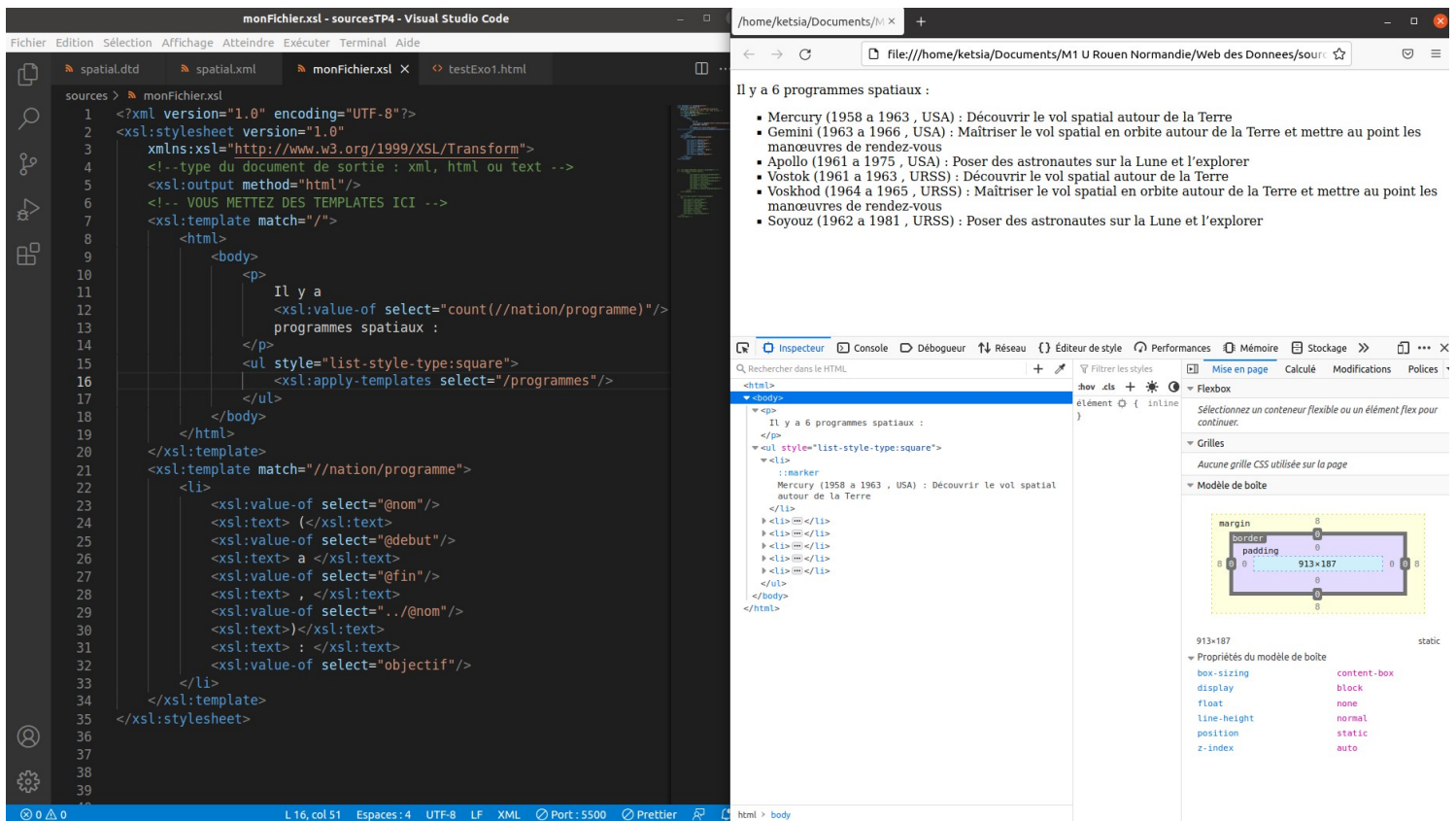
Pour se faire, on se permet de réaliser cette question de 2 façons.

1) D'abord en utilisant une boucle « **for-each** » :

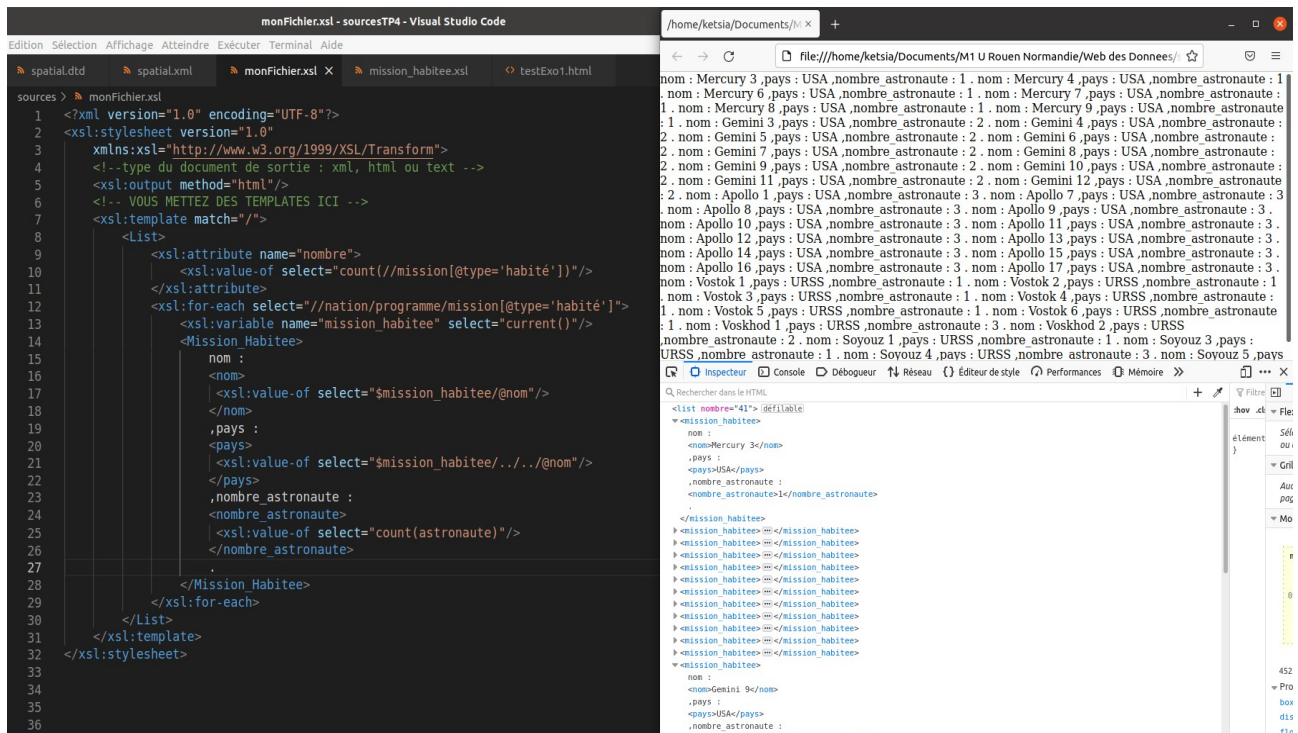
```
<xsl:template match="/">
<html>
<body>
<p>
Il y a
<xsl:value-of select="count(//nation/programme)"/>
programmes spatiaux :
</p>
<ul style="list-style-type:square">
<xsl:for-each select="//nation/programme">
<li>
<xsl:value-of select="@nom"/>
<xsl:text> (</xsl:text>
<xsl:value-of select="@debut"/>
<xsl:text> a </xsl:text>
<xsl:value-of select="@fin"/>
<xsl:text> , </xsl:text>
<xsl:value-of select="../@nom"/>
<xsl:text>)</xsl:text>
<xsl:text> : </xsl:text>
<xsl:value-of select="objectif"/>
</li>
</xsl:for-each>
</ul>
</body>
</html>
</xsl:template>
```



2) Sans utiliser une boucle for-each, mais plutôt, à l'aide d'une apply-template



2.2 Exercice 2



```
sources > monFichier.xsl
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsl:stylesheet version="1.0"
3   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4   <!-- type du document de sortie : xml, html ou text -->
5   <xsl:output method="xml"/>
6   <!-- VOUS METTEZ DES TEMPLATES ICI -->
7   <xsl:template match="/">
8     <List>
9       <xsl:attribute name="nombre">
10        <xsl:value-of select="count(//mission[@type='habité'])"/>
11      </xsl:attribute>
12      <xsl:for-each select="//nation/programme/mission[@type='habité']">
13        <xsl:variable name="mission_habitee" select="current()"/>
14        <Mission_Habitee>
15          nom :
16          <nom>
17            <xsl:value-of select="$mission_habitee/@nom"/>
18          </nom>
19          ,pays :
20          <pays>
21            <xsl:value-of select="$mission_habitee/../../@nom"/>
22          </pays>
23          ,nombre_astronaute :
24          <nombre_astronaute>
25            <xsl:value-of select="count(astronaute)"/>
26          </nombre_astronaute>
27        </Mission_Habitee>
28      </xsl:for-each>
29    </List>
30  </xsl:template>
31 </xsl:stylesheet>
```



```

<list nombre="41"> défilable
  <mission_habitee>
    nom :
    <nom>Mercury 3</nom>
    ,pays :
    <pays>USA</pays>
    ,nombre_astronaute :
    <nombre_astronaute>1</nombre_astronaute>
    .
  </mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>...</mission_habitee>
  <mission_habitee>
    nom :
    <nom>Gemini 9</nom>
    ,pays :
    <pays>USA</pays>
    ,nombre_astronaute :
    <nombre_astronaute>2</nombre_astronaute>
    .
  </mission_habitee>
  <mission_habitee>...</mission_habitee>

```

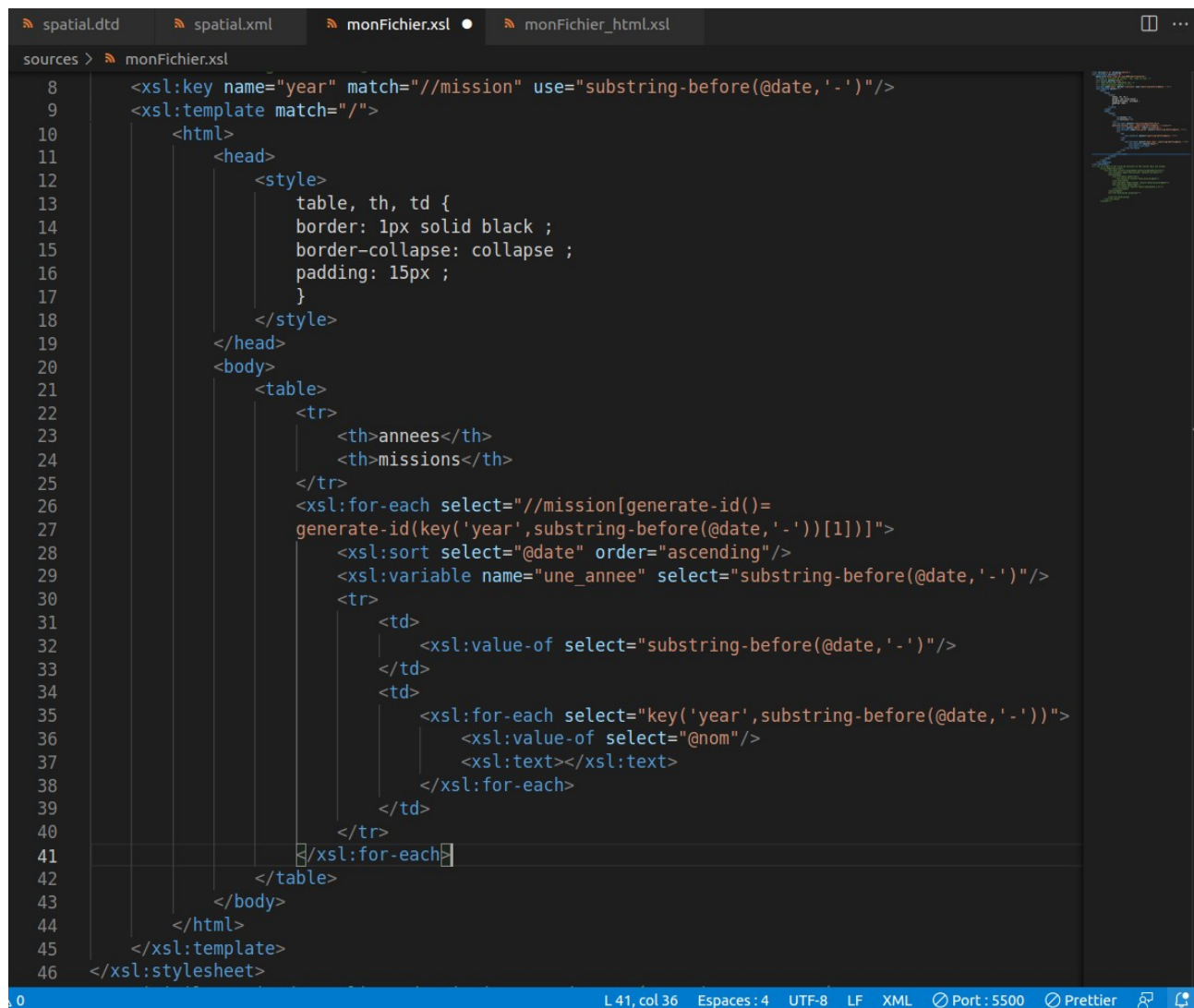
/home/ketsia/Documents/M1 U Rouen Normandie/Web des Donnees/sc

```

nom : Mercury 3 ,pays : USA ,nombre_astronaute : 1 . nom : Mercury 4 ,pays : USA ,nombre_astronaute : 1 .
nom : Mercury 6 ,pays : USA ,nombre_astronaute : 1 . nom : Mercury 7 ,pays : USA ,nombre_astronaute : 1 .
nom : Mercury 8 ,pays : USA ,nombre_astronaute : 1 . nom : Mercury 9 ,pays : USA ,nombre_astronaute : 1 .
nom : Gemini 3 ,pays : USA ,nombre_astronaute : 2 . nom : Gemini 4 ,pays : USA ,nombre_astronaute : 2 .
nom : Gemini 5 ,pays : USA ,nombre_astronaute : 2 . nom : Gemini 6 ,pays : USA ,nombre_astronaute : 2 .
nom : Gemini 7 ,pays : USA ,nombre_astronaute : 2 . nom : Gemini 8 ,pays : USA ,nombre_astronaute : 2 .
nom : Gemini 9 ,pays : USA ,nombre_astronaute : 2 . nom : Gemini 10 ,pays : USA ,nombre_astronaute : 2 .
nom : Gemini 11 ,pays : USA ,nombre_astronaute : 2 . nom : Gemini 12 ,pays : USA ,nombre_astronaute : 2 .
nom : Apollo 1 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 7 ,pays : USA ,nombre_astronaute : 3 .
nom : Apollo 8 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 9 ,pays : USA ,nombre_astronaute : 3 .
nom : Apollo 10 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 11 ,pays : USA ,nombre_astronaute : 3 .
nom : Apollo 12 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 13 ,pays : USA ,nombre_astronaute : 3 .
nom : Apollo 14 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 15 ,pays : USA ,nombre_astronaute : 3 .
nom : Apollo 16 ,pays : USA ,nombre_astronaute : 3 . nom : Apollo 17 ,pays : USA ,nombre_astronaute : 3 .
nom : Vostok 1 ,pays : URSS ,nombre_astronaute : 1 . nom : Vostok 2 ,pays : URSS ,nombre_astronaute : 1 .
nom : Vostok 3 ,pays : URSS ,nombre_astronaute : 1 . nom : Vostok 4 ,pays : URSS ,nombre_astronaute : 1 .
nom : Vostok 5 ,pays : URSS ,nombre_astronaute : 1 . nom : Vostok 6 ,pays : URSS ,nombre_astronaute : 1 .
nom : Voskhod 1 ,pays : URSS ,nombre_astronaute : 3 . nom : Voskhod 2 ,pays : URSS ,nombre_astronaute :
2 . nom : Soyouz 1 ,pays : URSS ,nombre_astronaute : 1 . nom : Soyouz 3 ,pays : URSS ,nombre_astronaute :
1 . nom : Soyouz 4 ,pays : URSS ,nombre_astronaute : 3 . nom : Soyouz 5 ,pays : URSS ,nombre_astronaute :
3 . nom : Soyouz 6 ,pays : URSS ,nombre_astronaute : 2 .

```

2.3 Exercice 3



```
8 <xsl:key name="year" match="//mission" use="substring-before(@date, '-')"/>
9 <xsl:template match="/">
10   <html>
11     <head>
12       <style>
13         table, th, td {
14           border: 1px solid black ;
15           border-collapse: collapse ;
16           padding: 15px ;
17         }
18       </style>
19     </head>
20     <body>
21       <table>
22         <tr>
23           <th>annees</th>
24           <th>missions</th>
25         </tr>
26         <xsl:for-each select="//mission[generate-id()=
27           generate-id(key('year', substring-before(@date, '-'))[1])]">
28           <xsl:sort select="@date" order="ascending"/>
29           <xsl:variable name="une_annee" select="substring-before(@date, '-')"/>
30           <tr>
31             <td>
32               <xsl:value-of select="substring-before(@date, '-')"/>
33             </td>
34             <td>
35               <xsl:for-each select="key('year', substring-before(@date, '-'))">
36                 <xsl:value-of select="@nom"/>
37                 <xsl:text></xsl:text>
38               </xsl:for-each>
39             </td>
40           </tr>
41         </xsl:for-each>
42       </table>
43     </body>
44   </html>
45 </xsl:template>
46 </xsl:stylesheet>
```

0 L 41, col 36 Espaces : 4 UTF-8 LF XML Port : 5500 Prettier

Rechercher dans le HTML

<html> défilable

> <head> ... </head>

▼ <body>

▼ <table> débordement

▼ <tbody>

▼ <tr>

<th>annees</th>

<th>missions</th>

</tr>

▼ <tr>

<td>1961</td>

<td>Mercury 3Mercury 4Vostok 1Vostok 2</td>

</tr>

▼ <tr>

<td>1962</td>

<td>Mercury 6Mercury 7Mercury 8Vostok 3Vostok 4</td>

</tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

▼ <tr>

<td>1965</td> débordement

▼ <td> débordement

Gemini 2Gemini 3Gemini 4Gemini 5Gemini 6Gemini 7Voskhod 2

</td>

</tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

▶ <tr> ... </tr>

</tbody>

</table>

</body>

</html>

annees	missions
1961	Mercury 3Mercury 4Vostok 1Vostok 2
1962	Mercury 6Mercury 7Mercury 8Vostok 3Vostok 4
1963	Mercury 9Vostok 5Vostok 6
1964	Gemini 1Voskhod 1
1965	Gemini 2Gemini 3Gemini 4Gemini 5Gemini 6Gemini 7Voskhod 2
1966	Gemini 8Gemini 9Gemini 10Gemini 11Gemini 12
1967	Apollo 1Apollo 4Soyouz 1
1968	Apollo 5Apollo 6Apollo 7Apollo 8Soyouz 2Soyouz 3
1969	Apollo 9Apollo 10Apollo 11Apollo 12Soyouz 4Soyouz 5Soyouz 6
1970	Apollo 13
1971	Apollo 14Apollo 15
1972	Apollo 16Apollo 17