ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №7

XSLT - мова перетворення XML-документів

Виконав:

Ст. Прізвище Ім’я

Група ……..

2022

Тема: Вивчення мови XSLT - мови перетворення XML-документів.

Мета роботи: Ознайомлення з синтаксисом мови XSLT та перетворенням фрагмента XML документа з використанням XSLT.

Теоретичний матеріал:

XSLT (XSL Transformations) це мова, призначена для перетворення XML-документів у документи іншого формату, наприклад HTML. Вона використовується спільно з XPath і поділяє ту саму модель даних, що і XPath. XSLT також використовує бібліотеку функцій та операторів, визначені в *XQuery and XPath Functions and Operators*. Мова XSLT визначена W3C в **XSL** Transformations (XSLT) Version 3.0 W3C Recommendation 8 June 2017

Як і XPath, XSLT є підмножиною XSL (eXtensible Stylesheet Language) - мова стилів таблиць для XML.

HTML використовує попередньо визначені теги. Значення та відображення кожного тегу добре зрозуміло. Для додавання стилів до елементів HTML використовуються каскадні таблиці стилів CSS.

XSL це фактично таблиці стилів для XML. XML не використовує попередньо визначені теги, і тому значення кожного тегу комп'ютеру наперед невідомо (наприклад, елемент <table> може вказувати таблицю HTML, предмет меблів чи щось інше - і браузери не знають, як це відобразити). Отже, XSL описує, як повинні відображатися елементи XML. Але XSL - більше, ніж просто мова таблиці стилів.

XSL складається з чотирьох частин:

XSLT - мова для перетворення XML-документів
XPath - мова для навігації в XML-документах
XSL-FO - мова для форматування XML-документів (завдяки модулю CSS3 Paged Media Module, W3C представив новий стандарт форматування документа і з 2013 року CSS3 пропонується як заміна XSL-FO)
XQuery - мова для запитів XML-документів

Перетворення мовою XSLT виражається у вигляді таблиці стилів. Таблиця стилів складається з одного або декількох добре сформованих XML документів, що відповідають просторам імен у Рекомендації XML.

Таблиця стилів, як правило, включає елементи, визначені XSLT, а також елементи, які не визначені XSLT. Елементи, визначені XSLT, розрізняються за допомогою простору імен http://www.w3.org/1999/XSL/Transform (див. 3.1 Простір імен XSLT. Таким чином, ця специфікація є визначенням синтаксису та семантики простору імен XSLT.

Термін таблиця стилів відображає той факт, що одна з важливих ролей XSLT полягає в додаванні інформації про стилізацію до вихідного документа XML, перетворюючи її в документ, що складається з об'єктів форматування XSL, або в інший презентаційно-орієнтований формат, такий як HTML, XHTML або SVG. Однак XSLT використовується для широкого спектру завдань трансформації, а не виключно для форматування та презентаційних програм.

Перетворення, виражене XSLT, описує правила перетворення вхідних даних у вихідні дані. Усі входи та виходи будуть екземплярами моделі даних XDM. У найпростішому і найпоширенішому випадку вхід - це XML-документ, який називається деревом-джерелом,

а вихід - документом XML, який називається деревом результатів. Можлива також обробка декількох вихідних документів, генерування декількох результативних документів та обробка форматів, відмінних від XML.

Перетворення досягається набором правил шаблону. Правило шаблону асоціює шаблон, який зазвичай відповідає вузлам у вихідному документі, із конструктором послідовностей. У багатьох випадках оцінка конструктора послідовностей призведе до побудови нових вузлів, які можна використовувати для отримання частини дерева результатів. Структура результативних дерев може бути абсолютно різною від структури дерев-джерел. При побудові дерева результатів вузли з дерев-джерел можуть бути відфільтровані та упорядковані, а також може бути додана довільна структура.

Такий механізм дозволяє застосовувати таблицю стилів для широкого класу документів, які мають схожі структури вихідного дерева. Таблиці стилів мають модульну структуру; вони можуть містити кілька пакетів, розроблених незалежно один від одного, і кожен пакет може складатися з декількох модулів таблиць стилів.

Отже, з допомогою XSLT можна додавати або видаляти елементи та атрибути до вихідного файлу або з нього. Також можна переставляти та сортувати елементи, виконувати тести та приймати рішення щодо того, які елементи ховати та відображати, та багато іншого.

У процесі перетворення XSLT використовує XPath для навігації по елементах та атрибутах у документа XML щоб визначити ті частини документа, які повинні відповідати одному або більше заздалегідь заданим шаблонам. Коли знайдено збіг, XSLT перетворить відповідну частину вхідного документа в документ результату.

1. Основні поняття XSLT

Для ознайомлення з основними поняттями XSLT розглянемо приклад перетворення наступного документа XML ("cdcatalog.xml") у XHTML:

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
     <cd>
       <title>Hide your heart</title>
       <artist>Bonnie Tyler</artist>
       <country>UK</country>
       <company>CBS Records</company>
       <price>9.90</price>
       <year>1988</year>
     </cd>
...
</catalog>
```

Нехай створена нами таблиця стилів XSL ("cdcatalog.xsl") з шаблоном перетворення має наступний вигляд:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="/">
  <html>
    <body>
      <h2>My CD Collection</h2>
      <table border="1">
        <tr bgcolor="#9acd32">
          <th>Title</th>
          <th>Artist</th>
        </tr>
        <tr>
          <td>...</td>
          <td>...</td>
        </tr>
      </table>
    </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Щоб зв'язати таблицю стилів XSL з документом XML додамо посилання на таблицю стилів XSL до документа XML ("cdcatalog.xml"):

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="cdcatalog.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
...
</catalog>
```

1.1. Декларація стиля

Повернемося до файлу таблиці стилів XSL ("cdcatalog.xsl").

Оскільки таблиця стилів XSL є документом XML, він завжди починається з декларації XML: `<?xml version="1.0" encoding="UTF-8"?>`.

Кореневим елементом, який оголошує документ як таблицю стилю XSL, є *<xsl: stylesheet>* або *<xsl: transform>* - обидва повні синоніми і можна використовувати будь-який з них. Правильний спосіб оголошення таблиці стилю XSL відповідно до рекомендації W3C XSLT:

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

або:

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

Щоб отримати доступ до елементів, атрибутів та особливостей XSLT, ми повинні оголосити простір імен XSLT у верхній частині документа.

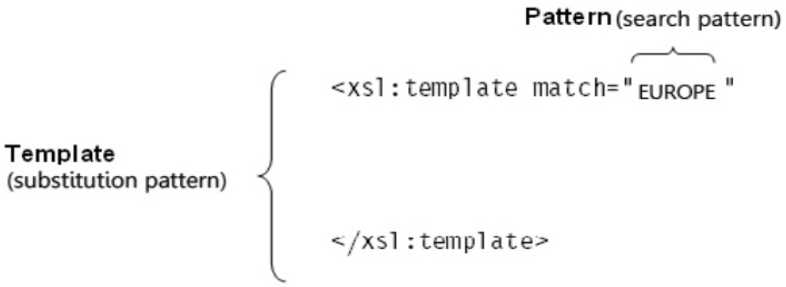
xmlns: xsl = "http://www.w3.org/1999/XSL/Transform" вказує на офіційний простір імен W3C XSLT. Якщо ви користуєтеся цим простором імен, ви також повинні включити атрибут *version = "1.0"*.

1.2. Шаблон перетворення

Таблиця стилів XSL складається з одного або декількох наборів правил, які називаються шаблонами. Правила шаблону - це правила для трансформації вихідного дерева в дерево результатів, тобто правила, які слід застосувати, коли вказаний вузол збігається .

Вони складаються з двох частин, одна з яких називається патерном (шаблон пошуку), а інша шаблоном (шаблон заміни), який виконується, коли шаблон пошуку має відповідний збіг у вихідному дереві.

Склад шаблону наступний



Для створення шаблонів використовується елемент <xsl: template>. В нашому прикладі він має вигляд <xsl:template match="/">. Атрибут match використовується для асоціації шаблону з XML-елементом. В даному випадку *match="/"* є виразом XPath, що пов'язує шаблон з коренем вихідного документа XML, тобто визначає весь документ.

Але в результаті використання наведеного шаблону перетворення жодні дані не будуть скопійовані з XML-документа на вихід. Ми отримаємо наступне:

My CD Collection

Title	Artist
…	…

1.3. Отримання значення XML-елемента <value-of>

Для того щоб отримати значення вибраного вузла та додати його до вихідного потоку перетворення потрібно використати елемент <xsl:value-of>. Так, якщо замість "…" у нашому прикладі записати

```
<xsl:value-of select="catalog/cd/title"/>
<xsl:value-of select="catalog/cd/artist"/>
```

отримаємо

Title	Artist
Empire Burlesque	Bob Dylan

Атрибут select містить вираз XPath і був застосований, щоб вибрати лише певну частину дерева. Усі інші піделементи таким чином ігноруються.

Таким чином ми отримали значення title і artist лише для одного вузла CD.

1.4. Цикл у шаблоні перетворення <for-each>

Елемент <xsl: for-each> дозволяє робити цикл у шаблоні перетворення і може використовуватися для вибору кожного XML-елемента певного набору вузлів. Це робиться наступним чином:

```
<?xml version="1.0" encoding="UTF-8"?>
 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="/">
   <html>
    <body>
     <h2>My CD Collection</h2>
      <table border="1">
       <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>
       </tr>
        <xsl:for-each  select="catalog/cd">
         <tr>
          <td><xsl:value-of  select="title | year"/>  <br /> </td>
          <td><xsl:value-of  select="artist"/> </td>
         </tr>
        </xsl:for-each>
      </table>
     </body>
    </html>
  </xsl:template>

 </xsl:stylesheet>
```

1.5. Сортування результатів виводу <sort>

Щоб сортувати вихід за прізвищами артистів, додамо елемент <xsl:sort> всередину елемента <xsl: for-every> у файл XSL:

```
<xsl:for-each select="catalog/cd">
  <xsl:sort select="artist"/>
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
 </xsl:for-each>
```

1.6. Вивід за умовою <if>

Елемент <xsl: if> використовується для встановлення умови виводу вмісту файлу XML. Наприклад, щоб вивести лише назви та виконавця компакт-дисків, ціна яких перевищує 10 потрібно записати наступний код

```
<xsl:for-each select="catalog/cd">
   <xsl:if test="price &gt; 10">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
      <td><xsl:value-of select="price"/></td>
    </tr>
   </xsl:if>
 </xsl:for-each>
```

1.7. Вивід за однією з кількох умов <choose>

Щоб вивести за однією з кількох умов у файл XML, потрібно скористатися елементом <xsl:choose> разом з <xsl: when> та <xsl:otherwise>:

```
<xsl:for-each select="catalog/cd">
   <tr>
     <td><xsl:value-of select="title"/></td>
     <xsl:choose>
       <xsl:when test="price &gt; 10">
         <td bgcolor="#ff00ff">
         <xsl:value-of select="artist"/></td>
       </xsl:when>
       <xsl:otherwise>
         <td><xsl:value-of select="artist"/></td>
       </xsl:otherwise>
     </xsl:choose>
   </tr>
   </xsl:for-each>
```

Наведений вище код додасть рожевий фоновий колір у стовпчик «Виконавець», коли ціна компакт-диска вище 10.

1.8. Правило шаблону до поточного елемента або дочірніх вузлів поточного елемента

Елемент <xsl: apply-templates> застосовує правило шаблону до поточного елемента або дочірніх вузлів поточного елемента.

Якщо ми додамо атрибут "select" до елемента <xsl: apply-templates>, він обробить лише дочірні елементи, які відповідають значенню атрибута. Також його можна використовувати щоб вказати в якому порядку повинні оброблятися дочірні вузли. Наприклад

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <xsl:apply-templates/>
  </body>
  </html>
</xsl:template>

<xsl:template match="cd">
  <p>
  <xsl:apply-templates select="title"/>
  <xsl:apply-templates select="artist"/>
  </p>
</xsl:template>

<xsl:template match="title">
  Title: <span style="color:#ff0000">
  <xsl:value-of select="."/></span>
  <br />
</xsl:template>

<xsl:template match="artist">
```

```
  Artist: <span style="color:#00ff00">
  <xsl:value-of select="."/></span>
  <br />
</xsl:template>

</xsl:stylesheet>
```

Результат використання шаблону:

My CD Collection

Title: Empire Burlesque
Artist: Bob Dylan

Title: Hide your heart
Artist: Bonnie Tyler

Title: Greatest Hits
Artist: Dolly Parton

1.9. Нумерація <number>

За допомогою елемента <xsl: number> вузли можуть бути призначені послідовностям у дереві результатів. Таким чином, наприклад, глави, розділи або прості елементи можуть бути пронумеровані. Також можлива нумерація в різних рівнях і може бути визначено форматування виводу. Нумерація арабськими або римськими номерами або літерами може бути сформована за допомогою атрибуту формату. На практиці елемент має три загальні ознаки: кількість (count) , рівень (level) та формат (format). Атрибут count визначає, які вузли потрібно підрахувати, рівень визначає спосіб підрахунку та формат визначає відображення. Наприклад у вище наведеному прикладі можна пронумерувати назви дисків наступним чином

```
<xsl:template match="cd">
  <p>
  <xsl:number format="1" level="any"/>
  <xsl:apply-templates select="title"/>
  <xsl:apply-templates select="artist"/>
  </p>
</xsl:template>
```

На виході отримаємо

My CD Collection

1 Title: Empire Burlesque
Artist: Bob Dylan

2 Title: Hide your heart
Artist: Bonnie Tyler

3 Title: Greatest Hits
Artist: Dolly Parton

…

операція АБО декількох елементів може бути досягнута за допомогою "|"

```
<xsl:template match="cd">
  <p>
    <xsl:apply-templates select="title | year"/>  <br />
    <xsl:apply-templates select="artist"/>
  </p>
</xsl:template>
```

My CD Collection

Title: Empire Burlesque
1985
Artist: Bob Dylan

Посилання для ознайомлення з теоретичним матеріалом теми «XSLT - мова перетворення XML-документів»:

- Beginning XML, 5-edition © 2012 Joe Fawcett, Liam R.E. Quin, Danny Ayers
- XSLT Tutorial W3C

Перелік розділів та понять, з якими необхідно ознайомитись для виконання завдання лабораторної роботи:

1. Основні поняття XSLT
1.1.Декларація стиля
1.2.Шаблон перетворення
1.3.Отримання значення XML-елемента <value-of>
1.4.Цикл у шаблоні перетворення <for-each>
1.5.Сортування результатів виводу <sort >
1.6.Вивід за умовою <if>
1.7.Вивід за однією з кількох умов <choose>
1.8.Правило шаблону до поточного елемента або його дочірніх елементів
1.9.Нумерація <number>

Хід роботи

1. Опрацювати теоретичний матеріал.
2. Створити шаблон перетворення XSLT фрагмента свого XML документа з обов'язковим використанням <value-of>, <for-each>, <sort >, <if>/<choose>,<number>.
3. Використати довільний XSLT processor, наприклад freeformatter.com, для перевірки коректності роботи створеного шаблона перетворення.
4. Оформити звіт про виконання лабораторної роботи, який має містити:
   - титульну сторінку;
   - тему, мету та завдання лабораторної роботи;
   - короткий перелік та опис створеного шаблона перетворення;
   - навести скріни екрану з кодом шаблона та результатом перетворення.
5. Завантажити в канал "БД. Лабораторна робота" своєї команди в Teams.