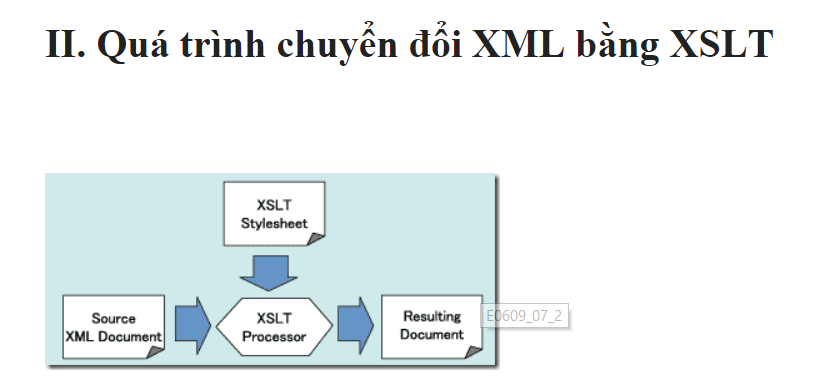
# 1. What is XSLT

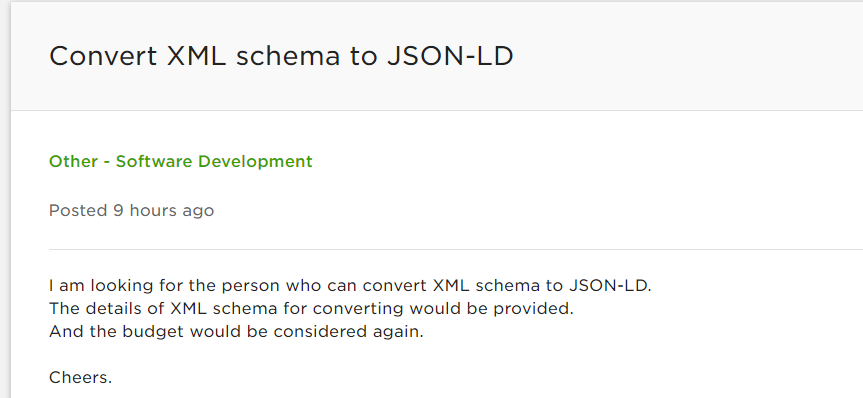
XSL is extensible stylesheet language

XSLT stands for XSL Transformations

<http://code5s.com/xml/xsl-v-xslt.html>



Có 1 công việc liên quan đến vấn đề xml schema này nọ đồ, ráng học đi làm việc về mấy cái này cũng có thêm tiền



<http://upworktestanswers.com/odesk-test-answers/internet-programming/xml-1_0-test-answers-2015.html>

**2. Which of the following attributes are optional in xsl:sort command?**

**Answers:**

• Select

• order

• datatype

• None of the above

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Value** | **Description** |
| select | XPath-expression | Optional. Specifies which node/node-set to sort on |
| lang | language-code | Optional. Specifies which language is to be used by the sort |
| data-type | text number qname | Optional. Specifies the data-type of the data to be sorted. Default is "text" |
| order | ascending descending | Optional. Specifies the sort order. Default is "ascending" |
| case-order | upper-first lower-first | Optional. Specifies whether upper- or lowercase letters are to be ordered first |

# 2. Stupid log for &lt

**&lt;**

The less-than character (<) starts **element markup** (the first character of a start-tag or an end-tag).

**&amp;**

The ampersand character (&) starts **entity markup** (the first character of a character entity reference).

**&gt;**

The greater-than character (>) ends a start-tag or an end-tag.

**&quot;**

The double-quote character (") can be symbolised with this character entity reference when you need to embed a double-quote inside a string which is already double-quoted.

**&apos;**

The apostrophe or single-quote character (') can be symbolised with this character entity reference when you need to embed a single-quote or apostrophe inside a string which is already single-quoted.

# Template match sử dụng xpath expression

<https://www.w3schools.com/xml/xpath_syntax.asp>

# Data type in xml

<https://msdn.microsoft.com/en-us/library/ms256220(v=vs.110).aspx>

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Facets** | **Description** |
| **string** | length, pattern, maxLength, minLength, enumeration, whiteSpace | Represents character strings. |
| **boolean** | pattern, whiteSpace | Represents Boolean values, which are either **true** or **false**. |
| **decimal** | enumeration, pattern, totalDigits, fractionDigits, minInclusive, maxInclusive, maxExclusive, whiteSpace | Represents arbitrary precision numbers. |
| **float** | pattern, enumeration, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents single-precision 32-bit floating-point numbers. |
| **double** | pattern, enumeration, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents double-precision 64-bit floating-point numbers. |
| **duration** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a duration of time.  The pattern for **duration** is PnYnMnDTnHnMnS, where nY represents the number of years, nM the number of months, nD the number of days, T the date/time separator, nH the number of hours, nM the number of minutes, and nS the number of seconds. |
| **dateTime** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a specific instance of time.  The pattern for **dateTime** is CCYY-MM-DDThh:mm:ss where CC represents the century, YY the year, MM the month, and DD the day, preceded by an optional leading negative (-) character to indicate a negative number. If the negative character is omitted, positive (+) is assumed. The T is the date/time separator and hh, mm, and ssrepresent hour, minute, and second respectively. Additional digits can be used to increase the precision of fractional seconds if desired. For example, the format ss.ss... with any number of digits after the decimal point is supported. The fractional seconds part is optional.  This representation may be immediately followed by a "Z" to indicate Coordinated Universal Time (UTC) or to indicate the time zone. For example, the difference between the local time and Coordinated Universal Time, immediately followed by a sign, + or -, followed by the difference from UTC represented as hh:mm (minutes is required). If the time zone is included, both hours and minutes must be present. |
| **time** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents an instance of time that recurs every day.  The pattern for **time** is hh:mm:ss.sss with optional time zone indicator. |
| **date** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a calendar date.  The pattern for **date** is CCYY-MM-DD with optional time zone indicator as allowed for **dateTime**. |
| **gYearMonth** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a specific Gregorian month in a specific Gregorian year. A set of one-month long, nonperiodic instances.  The pattern for **gYearMonth** is CCYY-MM with optional time zone indicator. |
| **gYear** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a Gregorian year. A set of one-year long, nonperiodic instances.  The pattern for **gYear** is CCYY with optional time zone indicator as allowed for **dateTime**. |
| **gMonthDay** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a specific Gregorian date that recurs, specifically a day of the year such as the third of May. A **gMonthDay** is the set of calendar dates. Specifically, it is a set of one-day long, annually periodic instances.  The pattern for **gMonthDay** is --MM-DD with optional time zone indicator as allowed for **date**. |
| **gDay** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a Gregorian day that recurs, specifically a day of the month such as the fifth day of the month. A **gDay** is the space of a set of calendar dates. Specifically, it is a set of one-day long, monthly periodic instances.  The pattern for **gDay** is ---DD with optional time zone indicator as allowed for **date**. |
| **gMonth** | enumeration, pattern, minInclusive, minExclusive, maxInclusive, maxExclusive, whiteSpace | Represents a Gregorian month that recurs every year. A **gMonth** is the space of a set of calendar months. Specifically, it is a set of one-month long, yearly periodic instances.  The pattern for **gMonth** is --MM-- with optional time zone indicator as allowed for **date**. |
| **hexBinary** | length, pattern, maxLength, minLength, enumeration, whiteSpace | Represents arbitrary hex-encoded binary data. A **hexBinary** is the set of finite-length sequences of binary octets. Each binary octet is encoded as a character tuple, consisting of two hexadecimal digits ([0-9a-fA-F]) representing the octet code. |
| **base64Binary** | length, pattern, maxLength, minLength, enumeration, whiteSpace | Represents Base64-encoded arbitrary binary data. A **base64Binary** is the set of finite-length sequences of binary octets. |
| **anyURI** | length, pattern, maxLength, minLength, enumeration, whiteSpace | Represents a URI as defined by RFC 2396. An **anyURI** value can be absolute or relative, and may have an optional fragment identifier. |
| **QName** | length, enumeration, pattern, maxLength, minLength, whiteSpace | Represents a qualified name. A qualified name is composed of a prefix and a local name separated by a colon. Both the prefix and local names must be an NCName. The prefix must be associated with a namespace URI reference, using a namespace declaration. |
| **NOTATION** | length, enumeration, pattern, maxLength, minLength, whiteSpace | Represents a **NOTATION** attribute type. A set of QNames. |

# Xml Schema

Validate online

<https://www.freeformatter.com/xml-validator-xsd.html>

## Restriction

<https://www.w3schools.com/xml/schema_facets.asp>

# DTD

Data type definition

<https://www.w3schools.com/xml/xml_dtd_intro.asp>

Validate DTD online:

<https://www.xmlvalidation.com/>

## DTD - XML Building Blocks

### Elements

Elements are the **main building blocks** of both XML and HTML documents.

Examples of HTML elements are "body" and "table". Examples of XML elements could be "note" and "message". Elements can contain text, other elements, or be empty. Examples of empty HTML elements are "hr", "br" and "img".

Examples:

<body>some text</body>  
  
<message>some text</message>

### Attributes

Attributes provide **extra information about elements**.

Attributes are always placed inside the opening tag of an element. Attributes always come in name/value pairs. The following "img" element has additional information about a source file:

<img src="computer.gif" />

The name of the element is "img". The name of the attribute is "src". The value of the attribute is "computer.gif". Since the element itself is empty it is closed by a " /".

### Entities

Some characters have a special meaning in XML, like the less than sign (<) that defines the start of an XML tag.

Most of you know the HTML entity: "&nbsp;". This "no-breaking-space" entity is used in HTML to insert an extra space in a document. Entities are expanded when a document is parsed by an XML parser.

The following entities are predefined in XML:

|  |  |
| --- | --- |
| **Entity References** | **Character** |
| &lt; | < |
| &gt; | > |
| &amp; | & |
| &quot; | " |
| &apos; | ' |

### CDATA

CDATA means character data.

**CDATA is text that will NOT be parsed by a parser**. Tags inside the text will NOT be treated as markup and entities will not be expanded.

https://freetuts.net/tim-hieu-cdata-trong-xml-519.html

1. CDATA trong XML

Một tài liệu XML được xây dựng bởi các thẻ XML do lập trình viên tự định nghĩa ra, và đương nhiên lập trình viên phải tuân thủ các quy tắc mà XML đề ra (xem bài [XML Element](https://freetuts.net/tim-hieu-element-trong-xml-515.html)). Tuy nhiên có một số trường hợp dữ liệu chứa trong các thẻ XML lại có những ký tự đặc biệt dẫn đến tài liệu XML bị đảo lộn.

**Ví dụ**:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <?xml version="1.0" encoding="UTF-8"?>  <Tutorials>      <Post>          <Title>              Học CDATA trong XML miễn phí          </Title>          <Code>              <?xml version="1.0" encoding="UTF-8"?>              <Persons>                  <Person>                     Nguyễn Văn Cường                  </Person>              </Persons>          </Code>      </Post>  </Tutorials> |

Bạn thấy nội dung bên trong thẻ Code là một đoạn mã XML khác, có nghĩa đây là trường hợp giá trị của the XML lại là một đoạn mã XML khác. Nếu chạy lên sẽ bị lỗi như hình dưới đây:



Để khắc phục trường hợp này ta sẽ sử dụng CDATA như sau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | <?xml version="1.0" encoding="UTF-8"?>  <Tutorials>      <Post>          <Title>              Học CDATA trong XML miễn phí          </Title>          <Code>              <![CDATA[              <?xml version="1.0" encoding="UTF-8"?>              <Persons>                  <Person>                     Nguyễn Văn Cường                  </Person>              </Persons>              ]]>          </Code>      </Post>  </Tutorials> |

Như vậy cú pháp của CDATA là <![CDATA[nội dung]]>.

2. Sử dụng Special Character trong XML

Special character (đôi lúc ta gọi là thực thể) là những ký tự đặc biệt được chuyển đổi thành mỗi dãy các chữ cái và dấu &;.

Như ở trong phần một ta sử dụng CDATA để khắc phục lỗi nội dung XML chứa kí tự đặc biệt. Nhưng cũng có một cách khác là bạn chuyển những ký tự đặc biệt đó sang Special characters như bảng dưới đây.

| **Thực thể** | **Mô tả** |
| --- | --- |
| &apos; | Dấu nháy đơn ' |
| &amp; | Dấu & |
| &gt; | Dấu > |
| &lt; | Dấu < |
| &quot; | Dáu nháy kép " |

**Ví dụ**: Viết lại ví dụ trên bằng cách sử dụng Special character.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <?xml version="1.0" encoding="UTF-8"?>  <Tutorials>      <Post>          <Title>              Học CDATA trong XML miễn phí          </Title>          <Code>              &lt;?xml version="1.0" encoding="UTF-8"?&gt;              &lt;Persons&gt;                  &lt;Person&gt;                     Nguyễn Văn Cường                  &lt;/Person&gt;              &lt;/Persons&gt;          </Code>      </Post>  </Tutorials> |

Chạy file XML này lên và mọi thứ đều ổn.

3. Lời kết

Cả hai cách trên đều dùng để xử lý nội dung bên trong thẻ XML, tuy nhiên chúng ta vẫn hay sử dụng CDATA bởi vì nó đơn giản nhưng lại rất hiệu quả.

Bài tiếp theo chúng ta sẽ tìm hiểu một số cách hiển thị XML trên trình duyệt.

**3. Which of the following are methods of formatting of XML documents?**

**Answers:**

• CSS

• XSLT

• DTD

• Schema

**4. Which of the following data types are allowed in xsl:sort command?**

**Answers:**

• Text

• Number

• Date

• None of the above

**5. Which of the following patterns are correct for 'select' attribute**

**of xsl:sort command?**

**Answers:**

• text() for any text

• node() for any node including root node and attribute node

• / for the root node

• @\* for any parent node

@\* là sử dụng cho any attribute