情報科学演習 資料 9
XSLT とロケーションパス

令和 6 年 6 月 17 日

目 次

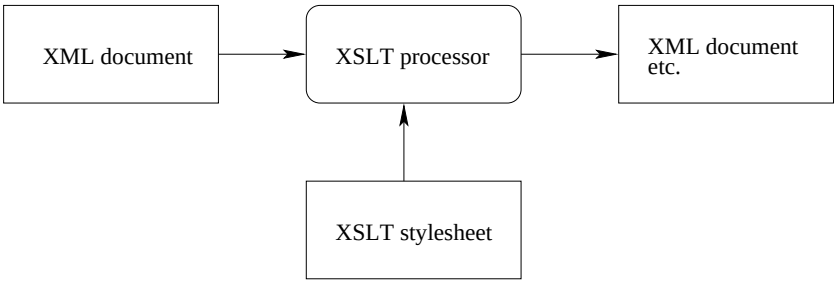
1 XSLT

この資料では XSLT (eXtensible Stylesheet Language Transformation) 言語の基礎と，XSLT を用いて XML 文書からデータの検索や抽出を行うための方法を学ぶ。

1.1 XSLT

XSLT は XML 文書を変換・加工するための言語の一つであり，XSLT を用いれば，XML 文書から必要なデータのみを取り出したり，XML 文書を Web ページ用の文書にする等のデータ操作ができる。

XSLT を利用するには，XML 文書をどのように変換するかを書いた XSLT スタイルシートを作り，XML 文書と XSLT スタイルシートを XSLT プロセッサで処理する。



2 XSLT における XML 文書のデータモデル

2.1 木構造データモデル

XSLT を使うには，変換処理やデータ抽出の対象となる XML 文書内の箇所や範囲を指定する必要がある。そのためには XSLT が XML 文書をどのように構造化して扱うか，すなわち XML 文書のデータモデルの理解が必要である。

図 1 の左側に XML 文書の一例を示す。この文書のルート要素は ref であり，ref の内容は book や web 等の要素である。

XSLT では，XML 文書を木 (tree) 構造のデータとして扱う。図 1 の右側は，左側の XML 文書を XSLT のデータモデル (木) として描画したものである。この木における / や ref などを**ノード (node)** と呼ぶ[1]。

この木には，次に記す種類のノードがある[2]。

ルートノード (root node) XML 文書全体に対応するノードであり / と表記される。ルートノード以外のすべてのノードは，ルートノードの子孫である。

[1] 木やノードはグラフ理論における用語であり，直感的には，ノード（節点）と節点同士をつなぐ辺 (edge) の集まりがグラフである。閉路を持たないグラフが木である。節点 (node) を頂点と呼び，辺 (edge) を枝と呼ぶこともある。

[2] 正確には，**テキストノード (text node)** という XML 文書の文字データに対応するノードも存在するが，省略した。図 1 の左側の XML 文書では，タグ (< >) の外にある Title1 等が文字データであり，これらに対応するのがテキストノードである。個々のテキストノードに名前は無く，テキストノードの文字データは親要素ノードが持つ文字列値として得られるため，テキストノードを意識する必要性は少ないと考えてよい。
XSLT で使われるノードには，この他にコメントノード，処理命令ノード，名前空間ノードがある。

```
<?xml version="1.0" ?>
<ref>
  <book isbn="1111">
    <title>Title1</title>
    <author>Author1</author>
  </book>
  <book>
    <title>Title2</title>
    <author>Author2</author>
    <year>2020</year>
  </book>
  <web>
    <title>Title3</title>
  </web>
</ref>
```

```
/
|
+- ref
     |
     +- book
     |   |
     |   +- @isbn (1111)
     |   |
     |   +- title (Title1)
     |   |
     |   +- author (Author1)
     |
     +- book
     |   |
     |   +- title (Title2)
     |   |
     |   +- author (Author2)
     |   |
     |   +- year (2020)
     |
     +- web
         |
         +- title (Title3)
```

図 1: XSLT における XML 文書のデータモデル (ノード表現)。右側の木において，/ はルートノードであり，@ で始まるノードは属性ノード，その他は要素ノードである。( ) 内は当該ノードの文字列値である。文字列値については後述する。

要素ノード (element node) XML 文書の各要素に対応するノード。XML 文書における「ルート要素」は，データモデルにおいてはルートノードの子である。個々の要素ノードの名前は XML 文書における要素名である。

属性ノード (attribute node) XML 文書の属性に対応するノード。属性ノードは要素ノードの子になる。個々の属性ノードの名前は XML 文書における属性名である。

2.2 ノードの指定方法: ロケーションパス

XML 文書をモデル化した木からノードを指定する記述が**ロケーションパス (location path)** である。この資料で扱うのはロケーションパスの省略記法のみであり，この場合，ロケーションパスの基本的な記述法はディレクトリやファイルのパス名と同じである。以下で示すロケーションパスの例は図 1 におけるものである。なお，ロケーションパスは XPath (XML Path Language) という規格で定められている[3]。

2.2.1 絶対ロケーションパス

絶対ロケーションパス (absolute location path) はルートノードを基点としてノードの位置を表現するものである。絶対ロケーションパスではルートノードを / で表記する。/ に続きルートノードから当該ノードに至る道筋に存在する要素ノードや属性ノードを / で区切って書き並べる

[3]XML 関係の仕様は https://www.w3.org/standards/xml/ から閲覧可能である。記述は英語だが，日本語訳が存在するものもある。

と，当該ノードのロケーションパスができあがる。その際，要素ノードと属性ノードは次のとおりに表記する。

要素ノードの表記法 ロケーションパスに要素ノードを含めるには，要素ノード名をそのまま使う。

例：/ref/web/title

属性ノードの表記法 ロケーションパス中に属性ノード記すには，ノード名の前に @ を付ける。

例：/ref/book/@isbn

2.2.2 ノードセット

ファイルやディレクトリのパス名では，パス名に対応するファイルやディレクトリがただ一つであるのに対し，XML のロケーションパスは一般にノードの集まり（**ノードセット; node-set**）である。例えば，図 1 の木における /ref/book は二つのノードからなるノードセットである。

ノードセットの指定には，次のようなロケーションパスを用いることもできる。

/ref/*	ref ノードのすべての子要素ノード
/ref/book/@*	book ノードのすべての子属性ノード
//title	ルートノードの子孫であるすべての title 要素ノード
/ref//title	ref ノードの子孫の中のすべての title 要素ノード
/ref/book|/ref/web	ref ノードの子要素ノード book と web をあわせたノードセット

2.2.3 ノードの絞り込み (filtering)

次の例のように，ロケーションパスに [...] を記述することで，ノードセットから ... の条件を満たすノード（またはノードセット）のみを指定したことになる[4]。

/ref/book[2]	2 番目の book 要素ノード
/ref/book[@isbn='1111']	isbn 属性が 1111 である book 要素ノード
/ref/book[year]	year 要素ノードを子に持つ book 要素ノード
/ref/book[2]/title	2 番目の book ノードの子である title 要素ノード

2.2.4 相対ロケーションパス

基点をルートノードに限定しないロケーションパスを**相対ロケーションパス (relative location path)** という。相対ロケーションパスでは，ディレクトリやファイルの相対パス名と同様に，ロケーションパスの始めに / を書かない。

相対ロケーションパスでは，基点となるノードを . （ドット）で表し，親ノードを .. （ドット二つ）で表す。

第 2.1 節の木において，仮りに web を基点とすれば，相対ロケーションパスの例は次のとおりとなる。

[4] [ ] の部分をロケーションパスの述部という

web の子ノードの title は ./title または title
ref は .. であり，/ は ../..
isbn は ../book/@isbn

相対ロケーションパスの実際の基点（= ドット . で表されるノード）は，XSLT では**カレントノード (current node)** と呼ばれる現在処理中のノードである[5]。カレントノードについては，次回の資料で触れる。

なお，相対ロケーションパスでも，第 2.2.3 節で紹介したノードの絞り込みが可能である。

2.2.5 xmllint コマンドを用いたロケーションパスの確認

次の形式で xmllint コマンドを実行すると，ロケーションパスで指定したノードに対応する XML 文書内の部分が出力される。

`xmllint --xpath` *expression file*

ここで *expression* にロケーションパスを与え[6]，*file* に XML 文書のファイル名を与える。

3 文字列値と XSLT スタイルシートにおける xsl:value-of 要素の使い方

この章では，ロケーションパスの使い方を確認することを目的として，そのために必要な最低限の XSLT スタイルシートの書き方を紹介する。

XSLT スタイルシートは，XML 文書をどのような形に変換するかを，XSLT で記したものである。なお，以下では XSLT スタイルシートを，単にスタイルシートと呼ぶことがある。

次の例は，変換対象となる XML 文書から，XML 宣言やタグ等を除いた文字データ部分をすべて出力するスタイルシートである。

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="text" encoding="UTF-8" />
  <xsl:template match="/">
    <xsl:value-of select="/" />
  </xsl:template>
</xsl:stylesheet>
```

XSLT スタイルシート自体も整形式の XML 文書なので，スタイルシートにも前回資料にある XML の用語（XML 宣言や要素，属性等）や規則が当てはまる。加えて XSLT 独自の規則等があるが，ここでは，ロケーションパスで指定されたノードの値（文字列値）を XML 文書から取り出すことができる xsl:value-of 要素に関わることのみを説明し，他の要素については次回の資料で説明する。

[5] 相対ロケーションパスの基点は一般には文脈ノード (context node) と呼ばれる。カレントノードは XSLT の用語であり，文脈ノードは XPath での用語。

[6] *expression* には XSLT の関数なども指定できる。また，シェルが *expression* 内の文字を特殊文字として解釈するのを防ぐためには，必要に応じて，*expression* を ’ ’ で囲む等するとよい。

3.1 文字列値

文字列値 (string-value) とは各ノードが持つ値である。文字列値は，ノードの種類毎に，次のとおりに定められている。

属性ノードの文字列値: 属性値

ルートノード，要素ノードの文字列値: ルートノードや要素ノードに対応する XML 文書の要素から，タグ<...> を除いたもの[7]

3.2 xsl:value-of 要素の使い方

xsl:value-of 要素は，XSLT スタイルシート内の xsl:template 要素（テンプレート）内で使える XSLT の要素であり，変換元の XML 文書から select 属性値で指定したロケーションパスの文字列値を取り出すことができる。

書式は次の通りである。

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

expression として，単一のノードに対応するロケーションパスを指定すれば，xsl:value-of はそのノードの文字列値を XML 文書から取り出す。複数のノード（ノードセット）に対応するロケーションパスを指定すれば，ノードセット中の最初のノードの文字列値のみを取り出す[8]。

例えば，先のスタイルシートにおける `<xsl:value-of select="/" />` は XML 文書からルートノード (/) の文字列値, すなわち XML 文書内の文字データ部分をすべて取り出すものである。

expression には，次の例のように，XSLT の関数も書ける。

```
<xsl:value-of select="count(/ref/book)" />
```

関数 count は，引数に与えられたノードセットのノード数を返すものであり，この xsl:value-of 要素を含むスタイルシートを図 1 の XML 文書に適用すれば /ref/book のノード数である 2 が出力される。

4 XSLT プロセッサ

XSLT を使うには，XSLT プロセッサと呼ばれるプログラムが必要である。これは単独のコマンドとして用意されていることや，多くの Web ブラウザのようにアプリケーションに組み込まれている場合がある。

ここでは，XSLT プロセッサとして xsltproc コマンドを用いる[9]。実行の書式は次のとおりである。

```
xsltproc [option] stylesheet file ...
```

ここで，*stylesheet* と *file* は，各々，XSLT スタイルシートと XML 文書の入ったファイルの名前である。例えば，

---

[7] 正確には，当該ノードの子孫となるテキストノードの文字列値を出現順に連結したもの
[8] XSLT バージョン 1 の場合
[9] xsltproc は XSLT バージョン 1 に準拠

```
xsltproc /pub/eis/xml/all.xsl /pub/eis/xml/ref.xml
```

は，XSLT スタイルシート /pub/eis/xml/all.xsl の内容に則って XML 文書 /pub/eis/xml/ref.xml を変換し，結果を標準出力（画面）に出力する。結果をファイルに入れたければ，リダイレクト (> *file*) を使えばよい。

5 問題

以下の問題では，まず，xmllint コマンドを使って，指定されたロケーションパス等に対応するノードが，設問で指示された XML 文書のどの部分なのかを確認しなさい。

また，xsltproc コマンドを使って，設問で指定された XML 文書に XSLT スタイルシートを適用して処理しなさい。XSLT スタイルシートとしては，第 3 章の XSLT スタイルシートの中の xsl:value-of 要素だけを変更して用いる。出力における空白や改行の乱れは無視してよい。

なお，第 3 章の XSLT スタイルシートは /pub/eis/xml/all.xsl として実習用コンピュータに置いてあるので，これを各自のホームディレクトリ配下のどこかにコピーするなどして使ってよい。

1. 第 3 章の XSLT スタイルシートにおいて，xsl:value-of 要素の select 属性値を以下の各ロケーションパスに変更し，xsltproc コマンドで図 1 の XML 文書 (/pub/eis/xml/data_model.xml) を処理すると，どのような出力が得られるかを考えなさい。実際に処理を行って結果を確認しなさい。

   (a) /ref/web/title

   (b) /ref/book/@isbn

   (c) /ref/book

   (d) /ref/book[2]

   (e) /ref/book[@isbn]

   (f) /ref/book[@isbn='1111']

   (g) /ref/book[2]/title

2. 任意の XML 文書から title 要素の総数を出力する XSLT スタイルシートを作成し，図 1 の XML 文書 (/pub/eis/xml/data_model.xml) および前回資料の XML 文書 (/pub/eis/xml/ref.xml) に適用しなさい。

3. 前回資料「XML の基礎」の「4 問題」の問 4 で作成した XML 文書から，自宅の郵便番号を出力する XSLT スタイルシートを作成しなさい。ただし，スタイルシートには，自宅の町名のロケーションパスを記述すること。