

# XML子要素の展開例1

2023-11-29

行方向の展開と列方向への展開サンプル2例組み込み

## 希望の結果

custommersの子要素群を行方向に、optsの子要素群を列方向に展開する。

この例だと、

- 要素recsの子要素であるrecの単位で出力する。
- 要素custommersの子要素であるcustomerの数だけ要素recの行数が増える。
- 要素optsの子要素であるoptの個々を独立した項目として扱う。最大3としたので項目数も3個追加された形になる。

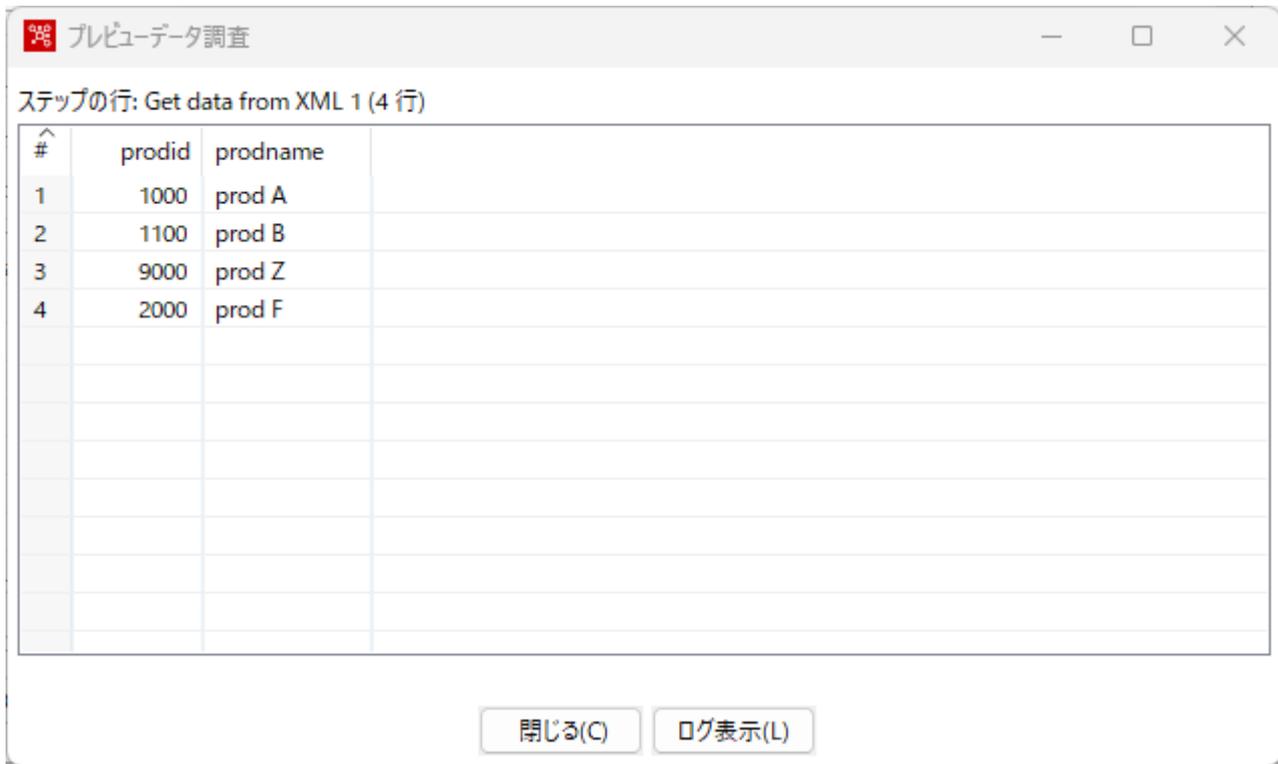
入力XML	出力(CSV想定)																																																																								
<pre> &lt;masterdata&gt;   &lt;recs&gt;     &lt;rec&gt;       &lt;prodid&gt;1000&lt;/prodid&gt;       &lt;prodname&gt;prod A&lt;/prodname&gt;       &lt;customers&gt;         &lt;customer&gt;&lt;name&gt;cust 01&lt;/name&gt;&lt;opts&gt;&lt;opt&gt;opt-A&lt;/opt&gt;&lt;/opts&gt;           &lt;addr&gt;addr 1&lt;/addr&gt;&lt;/customer&gt;         &lt;customer&gt;&lt;name&gt;cust 02&lt;/name&gt;&lt;opts&gt;&lt;opt&gt;opt-A&lt;/opt&gt; &lt;opt&gt;opt-B&lt;/opt&gt;&lt;/opts&gt; &lt;addr&gt;addr 2&lt;/addr&gt;&lt;/customer&gt;         &lt;customer&gt;&lt;name&gt;cust 10&lt;/name&gt;&lt;opts&gt;&lt;opt&gt;opt-A&lt;/opt&gt; &lt;opt&gt;opt-C&lt;/opt&gt;&lt;/opts&gt; &lt;addr&gt;addr 10&lt;/addr&gt;&lt;/customer&gt;       &lt;/customers&gt;     &lt;/rec&gt;     &lt;rec&gt;       &lt;prodid&gt;1100&lt;/prodid&gt;       &lt;prodname&gt;prod B&lt;/prodname&gt;       &lt;customers&gt;         &lt;customer&gt;&lt;name&gt;cust 02&lt;/name&gt;&lt;opts&gt;&lt;/opts&gt; &lt;addr&gt;addr 2&lt;/addr&gt;&lt;/customer&gt;         &lt;customer&gt;&lt;name&gt;cust 20&lt;/name&gt;&lt;addr&gt;addr 20&lt;/addr&gt;&lt;/customer&gt;       &lt;/customers&gt;     &lt;/rec&gt;     &lt;rec&gt;       &lt;prodid&gt;9000&lt;/prodid&gt;       &lt;prodname&gt;prod Z&lt;/prodname&gt;       &lt;customers&gt;         &lt;/customers&gt;     &lt;/rec&gt;     &lt;rec&gt;       &lt;prodid&gt;2000&lt;/prodid&gt;       &lt;prodname&gt;prod F&lt;/prodname&gt;       &lt;customers&gt;         &lt;customer&gt;&lt;name&gt;cust 52&lt;/name&gt;&lt;opts&gt;&lt;opt&gt;opt-A&lt;/opt&gt; &lt;opt&gt;opt-B&lt;/opt&gt; &lt;opt&gt;opt-C&lt;/opt&gt;&lt;/opts&gt; &lt;addr&gt;addr 52&lt;/addr&gt;&lt;/customer&gt;         &lt;customer&gt;&lt;name&gt;cust 55&lt;/name&gt;&lt;opts&gt;&lt;/opts&gt;&lt;/customer&gt;       &lt;/customers&gt;     &lt;/rec&gt;   &lt;/recs&gt; &lt;/masterdata&gt;           </pre>	<table border="1"> <thead> <tr> <th>prodid</th> <th>prodname</th> <th>customer</th> <th>opt1</th> <th>opt2</th> <th>opt3</th> <th>addr</th> </tr> </thead> <tbody> <tr><td>1000</td><td>prod A</td><td>cust 01</td><td>opt-A</td><td></td><td></td><td>addr 1</td></tr> <tr><td>1000</td><td>prod A</td><td>cust 02</td><td>opt-A</td><td>opt-B</td><td></td><td>addr 2</td></tr> <tr><td>1000</td><td>prod A</td><td>cust 10</td><td>opt-A</td><td>opt-C</td><td></td><td>addr 10</td></tr> <tr><td>1100</td><td>prod B</td><td>cust 02</td><td></td><td></td><td></td><td>addr 2</td></tr> <tr><td>1100</td><td>prod B</td><td>cust 20</td><td></td><td></td><td></td><td>addr 20</td></tr> <tr><td>9000</td><td>prod Z</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>2000</td><td>prod F</td><td>cust 52</td><td>opt-A</td><td>opt-B</td><td>opt-C</td><td>addr 52</td></tr> <tr><td>2000</td><td>prod F</td><td>cust 55</td><td></td><td></td><td></td><td></td></tr> </tbody> </table> <p>Get data from XML 1 → 行整列 1 → マージ結合 → 選択/名前変更 → ダミー (何もしない)</p> <p>Get data from XML 2 → 行整列 2 → マージ結合 → 選択/名前変更 → ダミー (何もしない)</p>	prodid	prodname	customer	opt1	opt2	opt3	addr	1000	prod A	cust 01	opt-A			addr 1	1000	prod A	cust 02	opt-A	opt-B		addr 2	1000	prod A	cust 10	opt-A	opt-C		addr 10	1100	prod B	cust 02				addr 2	1100	prod B	cust 20				addr 20	9000	prod Z						2000	prod F	cust 52	opt-A	opt-B	opt-C	addr 52	2000	prod F	cust 55													
prodid	prodname	customer	opt1	opt2	opt3	addr																																																																			
1000	prod A	cust 01	opt-A			addr 1																																																																			
1000	prod A	cust 02	opt-A	opt-B		addr 2																																																																			
1000	prod A	cust 10	opt-A	opt-C		addr 10																																																																			
1100	prod B	cust 02				addr 2																																																																			
1100	prod B	cust 20				addr 20																																																																			
9000	prod Z																																																																								
2000	prod F	cust 52	opt-A	opt-B	opt-C	addr 52																																																																			
2000	prod F	cust 55																																																																							
	<p><b>実行結果</b></p> <p>ログ   実行履歴   実行状況   パフォーマンスグラフ   メトリクス   データをプレビュー</p> <p>● 開始行 ○ 最終行 ○ Off</p> <table border="1"> <thead> <tr> <th>#</th> <th>prodid</th> <th>prodname</th> <th>customer</th> <th>opt1</th> <th>opt2</th> <th>opt3</th> <th>addr</th> </tr> </thead> <tbody> <tr><td>1</td><td>1000</td><td>prod A</td><td>cust 01</td><td>opt-A</td><td></td><td></td><td>addr 1</td></tr> <tr><td>2</td><td>1000</td><td>prod A</td><td>cust 02</td><td>opt-A</td><td>opt-B</td><td></td><td>addr 2</td></tr> <tr><td>3</td><td>1000</td><td>prod A</td><td>cust 10</td><td>opt-A</td><td>opt-C</td><td></td><td>addr 10</td></tr> <tr><td>4</td><td>1100</td><td>prod B</td><td>cust 02</td><td></td><td></td><td></td><td>addr 2</td></tr> <tr><td>5</td><td>1100</td><td>prod B</td><td>cust 20</td><td></td><td></td><td></td><td>addr 20</td></tr> <tr><td>6</td><td>2000</td><td>prod F</td><td>cust 52</td><td>opt-A</td><td>opt-B</td><td>opt-C</td><td>addr 52</td></tr> <tr><td>7</td><td>2000</td><td>prod F</td><td>cust 55</td><td></td><td></td><td></td><td></td></tr> <tr><td>8</td><td>9000</td><td>prod Z</td><td>&lt;null&gt;</td><td>&lt;null&gt;</td><td>&lt;null&gt;</td><td>&lt;null&gt;</td><td>&lt;null&gt;</td></tr> </tbody> </table>	#	prodid	prodname	customer	opt1	opt2	opt3	addr	1	1000	prod A	cust 01	opt-A			addr 1	2	1000	prod A	cust 02	opt-A	opt-B		addr 2	3	1000	prod A	cust 10	opt-A	opt-C		addr 10	4	1100	prod B	cust 02				addr 2	5	1100	prod B	cust 20				addr 20	6	2000	prod F	cust 52	opt-A	opt-B	opt-C	addr 52	7	2000	prod F	cust 55					8	9000	prod Z	<null>	<null>	<null>	<null>	<null>
#	prodid	prodname	customer	opt1	opt2	opt3	addr																																																																		
1	1000	prod A	cust 01	opt-A			addr 1																																																																		
2	1000	prod A	cust 02	opt-A	opt-B		addr 2																																																																		
3	1000	prod A	cust 10	opt-A	opt-C		addr 10																																																																		
4	1100	prod B	cust 02				addr 2																																																																		
5	1100	prod B	cust 20				addr 20																																																																		
6	2000	prod F	cust 52	opt-A	opt-B	opt-C	addr 52																																																																		
7	2000	prod F	cust 55																																																																						
8	9000	prod Z	<null>	<null>	<null>	<null>	<null>																																																																		

## 定義

### Get data from XML 1

要素recがレコードに変換されるので4レコード生成される事になる。





### 行整列 1



### Get data from XML 2



XPath指定の意のようで、相対パス指定等を利用できる。

- 要素customerのレコードに変換されるので7レコード生成される事になる。

- 親要素を辿ってprodidを取得。
- 要素optsの中身は添え字付きで指定し独立した項目として扱う。  
最大3個□COBOLでいうところのOCCURS句で3回の繰り返しとしている。

XMLデータ取得

ステップ名 Get data from XML 2

ファイル (全般 | フィールド | 追加出力フィールド)

設定

ループXPath /masterdata/recs/rec/custommers/customer XPathを取得

文字コード UTF-8

ドキュメントの名前空間を使用する

コメントを無視する

XMLを検証する

トークンを使用する

空のファイルを無視する

ファイルの有無を無視する

エラーを許容する回数 0

大容量データを処理するXPath

拡張設定

ファイル名を出力に含む  フィールド名

レコード番号を出力に含む  フィールド名

オプション設定

ファイル名を結果に含む

ヘルプ OK(O) プレビュー(V) キャンセル(C)

XMLデータ取得

ステップ名 Get data from XML 2

ファイル (全般 | フィールド | 追加出力フィールド)

#	フィールド名	XPath	要素タイプ	結果タイプ	データタイプ	書式	長さ	精度	通貨記号	桁区切り文字	数値囲み文字	空白除去	データを代替する
1	prodid	.././prodid	ノード	値	Integer							none	N
2	name	name	ノード	値	String							none	N
3	opt1	opts/opt[1]	ノード	値	String							none	N
4	opt2	opts/opt[2]	ノード	値	String							none	N
5	opt3	opts/opt[3]	ノード	値	String							none	N
6	addr	addr	ノード	値	String							none	N

フィールドの取得(G)

ヘルプ OK(O) プレビュー(V) キャンセル(C)

プレビューデータ調査

ステップの行: Get data from XML 2 (7 行)

#	prodid	name	opt1	opt2	opt3	addr
1	1000	cust 01	opt-A			addr 1
2	1000	cust 02	opt-A	opt-B		addr 2
3	1000	cust 10	opt-A	opt-C		addr 10
4	1100	cust 02				addr 2
5	1100	cust 20				addr 20
6	2000	cust 52	opt-A	opt-B	opt-C	addr 52
7	2000	cust 55				

閉じる(C) ログ表示(L)

## 行整列 2

行整列

ステップ名 行整列 2

一時ディレクトリ %%java.io.tmpdir%% 参照(B)...

一時ファイルの接頭子 out

並替えのサイズ (レコード数) 1000000

空きメモリのしきい値 (%)

一時ファイルを圧縮する

ユニークな行を対象にする

フィールド

#	フィールド名	昇順で並替え	大文字と小文字を区別する	Sort based on current locale?	Collator Strength	事前ソート
1	prodid	Y	N	N	0	N
2	name	Y	N	N	0	N

ヘルプ OK(O) キャンセル(C) フィールドを取得(G)

## マージ結合



Tipsにも記述があるけど入力には適切なソートが適用されている必要がある。

XML 1 LEFT OUTER JOIN XML 2 の意。キーはprodid

マージ結合

ステップ名

結合1のステップ名

結合2のステップ名

結合タイプ

フィールド (結合1)		フィールド (結合2)	
#	フィールド名	#	フィールド名
1	prodid	1	prodid

フィールドを取得(K)      フィールドを取得(V)

ヘルプ    OK(O)      キャンセル(C)

### 選択/名前変更

キー項目名が(この例だとprodid)が必ず使用される前提を付ける事ができるなら、マージ結合で生成される prodid\_1 は削除できる。

値選択

ステップ名

選択フィールド | 除去フィールド | メタ情報

#	フィールド名	変更フィールド名	データタイプ	長さ	精度	バイナリから変換	書式
1	name	customer	String			N	

フィールドの変更

ヘルプ      OK(O)      キャンセル(C)



[技術資料](#), [etl](#), [pentaho](#), [xml](#), [csv](#), [xpath](#)

From:  
<https://wiki.hgotoh.jp/> - 努力したWiki

Permanent link:  
<https://wiki.hgotoh.jp/documents/etl/pentaho/xml/pentaho-001>

Last update: **2025/11/20 08:52**

