DH ・ 技術要素 ・ XSLT / XPath

XSLT・XPath入門

XMLを選んで、作り変える

DH入門 ／ 技術要素シリーズ

中村

※実験的な取り組みです（構成・図・AI音声合成を含む）。内容はご確認・ご注意のうえご利用ください

この動画について

- ✓ **クリエイティブ・コモンズ**のオープン教材を参照し、独自に構成した解説です
- ✓ スライド・図は新規作成、ナレーションは**本人声のAI音声合成**
- ✓ これは**実験的な取り組み**です。内容は**ご確認・ご注意のうえ**ご利用ください
- ✓ 誤りに気づいたら概要欄からご指摘ください。出典・ライセンスは末尾と概要欄に記載しています

この回のゴール

XMLから目的の部分を「選び」、別の形へ「作り変える」流れをつかむ

- ✓ **XPath** で、XMLの木から目的のノードを**選ぶ**考え方を説明できる
- ✓ **XSLT** が、XMLを別の形（HTML等）へ**変換**する仕組みだと説明できる
- ✓ **テンプレート**（マッチ→出力）の基本の流れを説明できる
- ✓ TEI→Web公開など、DHでの使いどころと始め方を知る

TEI／XML入門を見ているとつながりますが、必須ではありません。

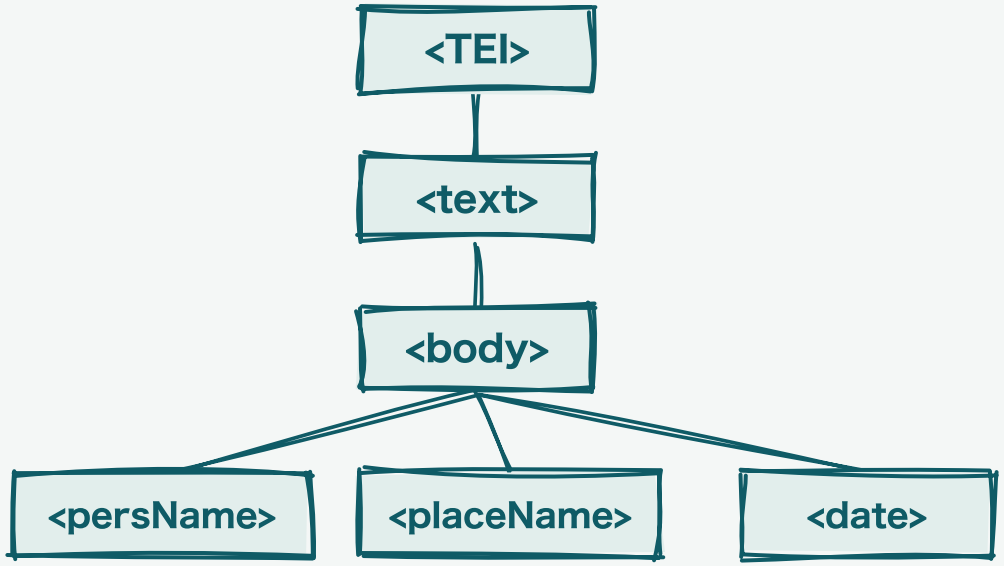
今日の流れ

- ✓ XMLは「中身」、見せ方は別もの
- ✓ 選ぶ — XPath
- ✓ 作り変える — XSLT（テンプレート）
- ✓ 全体の流れ（XML＋XSLT→出力）
- ✓ 使いどころ・始め方

XMLは「中身」、見せ方は別

まず、なぜ変換が要るのかから

XMLは「木（ツリー）」構造



XMLは、タグの入れ子＝「木（ツリー）」構造

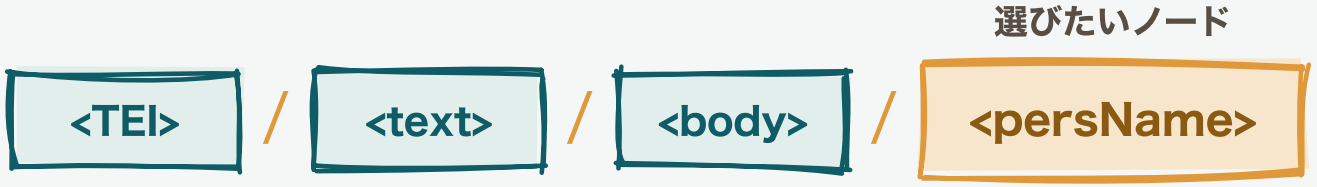
タグの**入れ子**でできた木。**persName**・**date**などの要素が枝葉に並ぶ

そのままでは「見せにくい」

- ✓ XMLは、構造をきちんと持つが、**人に見せる形**ではない
- ✓ Webページにしたい、一覧表にしたい、別の形式に移したい
- ✓ そのために、まず必要な部分を**選び**、つぎに**作り変える**
- ✓ 選ぶ道具が **XPath**、作り変える道具が **XSLT**

選ぶ — XPath

XPath＝木の「どこ」を指すパス



スラッシュで階層をたどり、目的のノードを選ぶ

（//persName と書けば「どこにあっても persName」を選ぶ）

フォルダのパスのように、**スラッシュで階層をたどって**ノードを指し示す

書き方：スラッシュでたどる

```
//persName
/TEI/text//date
```

// は「どこにあっても」。上は**人名を全部**、下は**本文中の日付**を選ぶ

角かっこ［ ］で条件を足す

```
//date[@when]
//persName[@role='author']
```

［ ］で絞り込み。「when属性を持つ日付」「役割が著者の人名」だけを選ぶ

ここまでのポイント

- ✓ XMLは、タグの入れ子でできた**木（ツリー）**
- ✓ **XPath** は、その木の「どこ」を指す**パス式**
- ✓ **//** でどこにあっても選び、**[ ]** で条件を足して絞る

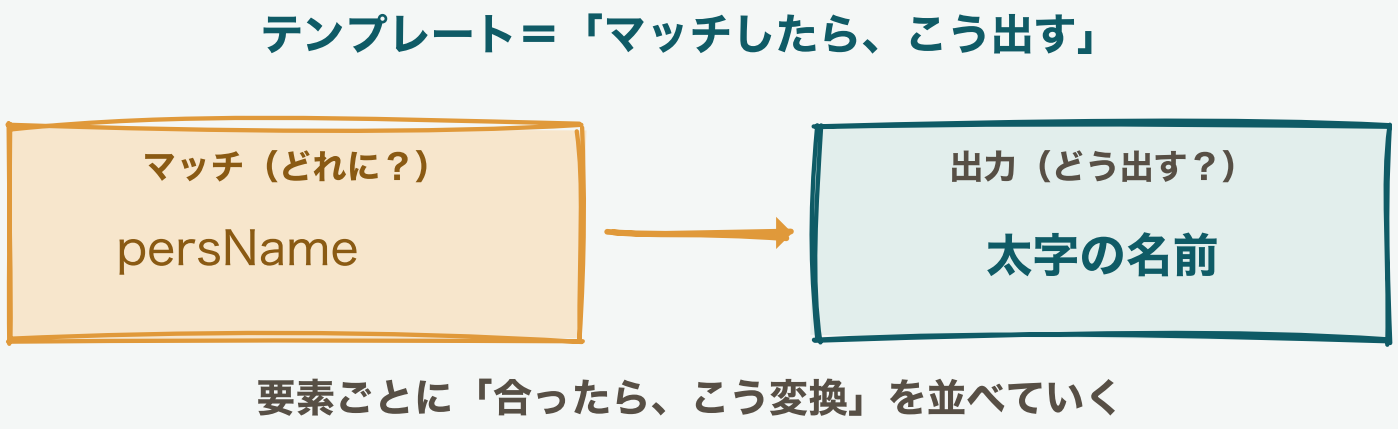
「選ぶ」道具がそろった。つぎは、選んだものを「作り変える」 → XSLT

作り変える — XSLT

XSLT＝変換の「指示書」

XSLT は、XMLを **HTML** など別の形へ作り変えるための指示書

テンプレート＝「マッチ→出力」



「**この要素に合ったら、こう出す**」という規則を、要素ごとに並べていく

テンプレートの書き方

```
<xsl:template match="persName">
  <strong><xsl:value-of select="."/></strong>
</xsl:template>
```

match で「どれに」、中身で「どう出すか」。persName を太字にして出す例

中の要素へ「流し込む」

```
<xsl:template match="body">
  <div><xsl:apply-templates/></div>
</xsl:template>
```

apply-templates で、中の要素を順に処理して**流し込む**

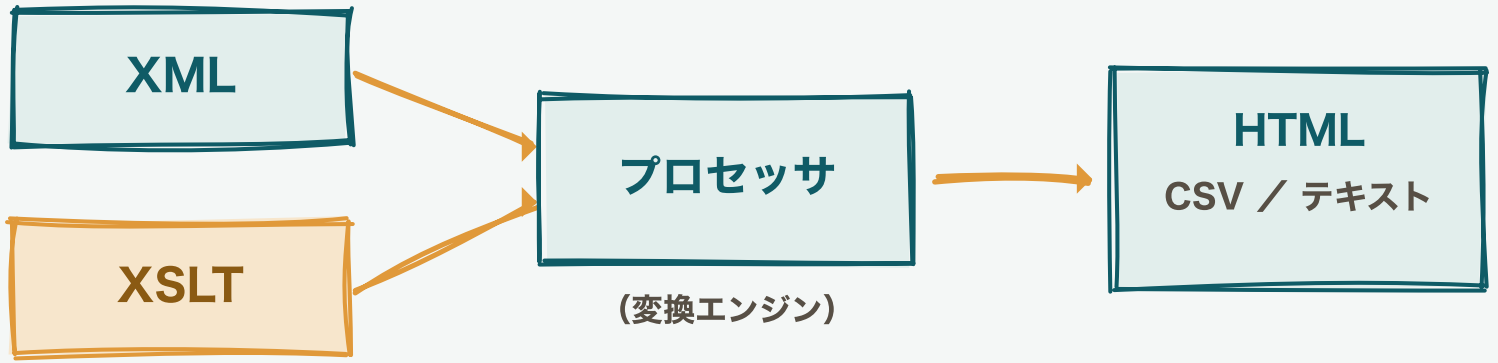
ここまでのポイント

- ✓ **XSLT** は、XMLを別の形へ作り変える**指示書**
- ✓ 基本単位は**テンプレート**＝「マッチ→出力」の規則
- ✓ **value-of** で中身を取り出し、**apply-templates** で中の要素へ流す

部品はそろった。XMLとXSLTを合わせると、どう動くのかを見る

全体の流れ

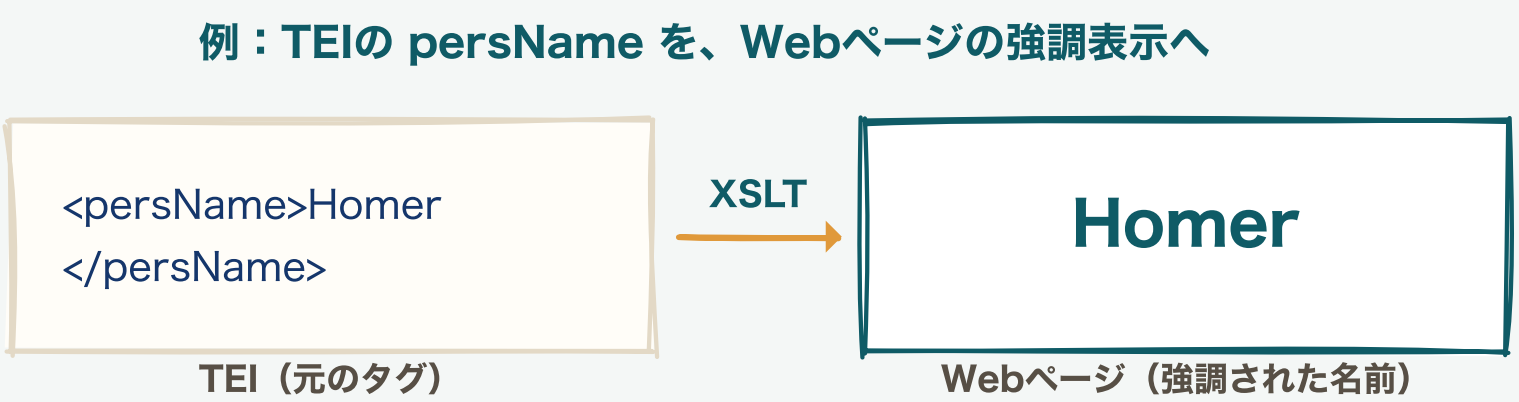
XML＋XSLT→プロセッサ→出力



XML と XSLT をプロセッサに渡すと、出力ができあがる

元の **XML** と指示書の **XSLT** を**プロセッサ**に渡すと、出力ができる

例：TEIをWebページへ



同じ中身でも、XSLTを変えれば**見た目や出力先を差し替え**られる

考えてみよう

あなたのXMLを、どんな形に**作り変え**たいですか？

Webページ？　一覧表？　別の形式？　ここで少し、動画を止めて考えてみてください。

ここまでのポイント

- ✓ 流れは **選ぶ（XPath）→ 作り変える（XSLT）→ 出力**
- ✓ XMLとXSLTを**プロセッサ**に渡すと、HTML等が得られる
- ✓ 中身はそのまま、**XSLTを差し替えれば**別の見せ方・出力にできる

使いどころ・始め方

DHでの活用

- ✓ **TEIをWeb公開**：符号化した本文を、読みやすいページに
- ✓ **一覧・索引づくり**：人名・地名・日付を抜き出して表に
- ✓ **別形式への書き出し**：プレーンテキストやCSVなどへ
- ✓ ひとつのXMLから、**複数の出力**を作り分けられる

「変換」と「問い合わせ」

- ✓ **XSLT** は、文書を**まるごと別の形へ作り変える**のが得意（変換）
- ✓ これに対し **XQuery** は、XMLから**条件で取り出す**のが得意（問い合わせ）
- ✓ どちらも内部で **XPath** を使う、同じ家族の技術
- ✓ 「ページを作る」ならXSLT、「データベースに問う」ならXQuery、と捉えると整理しやすい

RDFのグラフに問う **SPARQL** とは、対象とするデータの形が違います（XMLの木）

つまずきやすい点

- ✓ XSLT・XPathには**バージョン**がある（1.0／2.0／3.0）。書ける関数が違う
- ✓ 動かすには**プロセッサ**が要る（Saxon など。エディタに同梱のことも）
- ✓ まずは**小さな例**で、1つの要素を変換するところから
- ✓ うまく出ないときは、**XPathが正しく選べているか**を先に確かめる

始め方・学ぶには

- ✓ 体系的に：**Programming Historian**「Transforming Data ... with XML and XSL」
- ✓ 試す：**oXygen** などのエディタ、または **Saxon** で変換を実行
- ✓ TEI文脈：**TEI by Example** の XPath／XSLT モジュール
- ✓ コツ：**XPathで選ぶ → 1要素だけ変換**、を小さくくり返して育てる

まとめ

- ✓ XMLは**木構造**。**XPath** でその「どこ」を選ぶ
- ✓ **XSLT** は、選んだものを別の形へ**変換**する指示書
- ✓ 単位は**テンプレート**（マッチ→出力）。XML＋XSLT→プロセッサ→出力
- ✓ TEI→Web公開など、**1つのXMLから複数の出力**を作り分けられる

TEIで「タグを付けた」ものを、こんどは「選んで・作り変える」。地続きの考え方です

出典・ライセンス

本動画の**スライド・図・ナレーション原稿は CC BY 4.0** で公開します（© 2026 中村 覚）。出典表示のうえ自由に再利用いただけます。

- ✓ 参照教材：Transforming Data for Reuse and Re-publication with XML and XSL ／ M. H. Beals（Programming Historian）— CC BY 4.0

XPath／XSLTの仕様は W3C 勧告で事実確認（翻案せず）。掛け合い版の音声・立ち絵は VOICEVOX／坂本アヒル氏の各規約に従います。

ご清聴ありがとうございました