

# Język XSL

Tomasz Traczyk  
ttraczyk@ia.pw.edu.pl



Politechnika Warszawska  
Wydział Elektroniki i Technik Informacyjnych

# Język XSL

- > Co to jest XSL
- > Podstawy XSL
- > XPath
- > XSLT
- > Formatowanie
- > Zastosowania XSL
- > Podsumowanie



Język XML w aplikacjach z bazami danych

2

## Wprowadzenie

### Co to jest XSL (eXtensible Stylesheet Language)

- Uzupełnienie XML
- Służy do formatowania dokumentów XML — arkusze stylistyczne
- Może służyć do transformacji dokumentów XML i HTML

### Motywacja

- XML nie zawiera żadnych informacji o formatowaniu — znakowanie znaczeniowe
- Informacja o formatowaniu musi być zawarta w arkuszach stylistycznych

### Geneza

- Idea zaczerpnięta z DSSSL (*Document Style Semantics and Specification Language*)
- Język zdefiniowany w XML z użyciem przestrzeni nazw (*namespaces*)



Język XML w aplikacjach z bazami danych

3

### Co to jest XSL

- > Podstawy XSL
- > XPath
- > XSLT
- > Formatowanie
- > Zastosowania XSL
- > Podsumowanie



Język XML w aplikacjach z bazami danych

4

## Działanie XSL

### Zasada formatowania za pomocą XSL

- Transformacja drzewa znaczników na drzewo obiektów formatujących
- Transformacja steruje skrypt w XSL
- Obiekty formatujące mają przypisane sposoby prezentacji

### Opis transformacji

- Transformacja sterująca szablonami (*template rules*)
- Atrybut *match* szablonu
  - zawiera wzorec w języku XPath
  - określa, które elementy drzewa wyjściowego są przetwarzane na podstawie szablonu
- Zawartość szablonu
  - określa, co ma znaleźć się w drzewie wyjściowym
  - może zawierać rekurencyjne wywołania szablonów



Język XML w aplikacjach z bazami danych

5

## Fazy działania XSL

### Fazy działania XSL

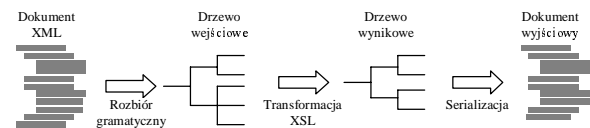
- Rozbiór gramatyczny (*parsing*) dokumentu wyjściowego
- Transformacja XSL
- Serializacja

### Możliwe postaci wyjściowe

- Drzewo znaczników (np. DOM)
- Drzewo obiektów formatujących
- Dokumenty (po serializacji)
  - XML
  - HTML
  - tekst

### Warunek przekształcenia

- Dokument wejściowy *well-formed*



Język XML w aplikacjach z bazami danych

6

## Przykład

### Dokument

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<!DOCTYPE eres_konsepky SYSTEM "konsepky.dtd">
<?xml-stylesheet type="text/xsl" href="konsepky.xslt"?>
<eres_konsepky>
  <przedmiot id="RBD2" wersja="1">
    <slowo_kluczowe>Bazy danych</slowo_kluczowe>
    <slowo_kluczowe>Oracle</slowo_kluczowe>
    <konsept>
      <czesc_konsepku id="Streszczenie">
        <P> Monograficzny przedmiot poświęcony bazie danych i narzędziom
        Oracle. </P>
      </czesc_konsepku>
      <czesc_konsepku id="Treść">
        <P> Omawiane są podstawowe zagadnienia związane z wykorzystaniem
        RDBMS Oracle7 i <ID>Oracle</ID> oraz administrowaniem nimi.</P>
        <P> Przedstawiane są także narzędzia do budowy aplikacji: </P>
        <UL>
          <LI> Oracle Forms. </LI>
          <LI> Oracle Reports. </LI>
        </UL>
      </czesc_konsepku>
    </konsept>
  </przedmiot>
</eres_konsepky>
```

## Przykład, c.d.

### Arkusz stylistyczny w XSL

```
<?xml version="1.0" encoding="windows-1250"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" indent="yes" encoding="windows-1250" />
  <xsl:template match="eres_konsepky">
    <xsl:comment>Wygenerowano za pomocą skryptu koncepky.xslt</xsl:comment>
    <HTML>
      <HEAD><TITLE>Konspekty</TITLE></HEAD>
      <BODY>
        <H1>Konspekty przedmiotów</H1>
        <xsl:apply-templates select="przedmiot">
          <xsl:sort select="id" />
        </xsl:apply-templates>
      </BODY>
    </HTML>
  </xsl:template>
```

## Przykład, c.d.

### Arkusz stylistyczny w XSL, c.d.

```
<xsl:template match="przedmiot">
  <H2>
    <xsl:value-of select="id" />
    |<xsl:value-of select="wersja" />
  </H2>
  <xsl:if test="slowo_kluczowe">
    <H3>Słowa kluczowe</H3>
    <TABLE BORDER="1">
      <xsl:for-each select="slowo_kluczowe">
        <xsl:sort select="text" />
        <TR><TD><xsl:value-of select="." /></TD></TR>
      </xsl:for-each>
    </TABLE>
  </xsl:if>
  <xsl:apply-templates select="*[@name() != 'slowo_kluczowe']"/>
</H2>
</xsl:template>

<xsl:template match="konsept">
  <H3>Konspekt</H3>
  <xsl:apply-templates />
</xsl:template>
```

## Przykład, c.d.

### Arkusz stylistyczny w XSL, c.d.

```
<xsl:template match="czesc_konsepku">
  <H4>
    <xsl:value-of select="id" />
  </H4>
  <xsl:apply-templates />
</xsl:template>

<xsl:template match="P | I | UL | OL | LI">
  <xsl:element name="{name()}">
    <xsl:apply-templates />
  </xsl:element>
</xsl:template>

</xsl:stylesheet>
```

## Przykład, c.d.

### Wynik przetwarzania



```
<!--Wygenerowano za pomocą skryptu
koncepky.xslt-->
<HTML>
  <HEAD>
    <META http-equiv="Content-Type"
    content="text/html;
    charset=windows-1250">
    <TITLE>Konspekty</TITLE>
  </HEAD>
  <BODY>
    <H1>Konspekty przedmiotów</H1>
    .....
  </BODY>
</HTML>
```

## Składnia

### XSL a XML

- XSL jest zdefiniowany w XML
- Ogólne zasady jak w XML (np. komentarze)
- Skrypt XSL musi być poprawny w sensie *valid*
- Dla stron kodowych różnych od UTF-8 obowiązkowa instrukcja `<?XML... ?>` z atrybutem `encoding`

### Owołanie do arkusza stylistycznego

- W dokumencie XML umieszczają się instrukcje `<?xml-stylesheet... ?>`

- Można umieścić kilka instrukcji dla różnych mediów, rozróżnionych atrybutem `media`

### Element główny arkusza

- Element główny `<xsl:stylesheet... />`
- Odwołanie do przestrzeni nazw `xsl`
- URI przestrzeni nazw decyduje o typie specyfikacji języka: XSL z roku 1998 czy XSLT

### Sterowanie serializacją

- Instrukcja `xsl:output` określa sposób serializacji
  - metodę
  - wcięcie
  - stronę kodową
  - prolog
- Metody serializacji
  - `xml` — wyprowadza dokument XML poprawny przynajmniej w sensie *well-formed*
  - `html` — wyprowadza dokument w HTML
  - `text` — wyprowadza zawartość tekstową elementów, ale bez znaczników

### „Ciało” arkusza XSL

- Zawiera szablon sterujący przetwarzaniem
- Może zawierać definicje zmiennych globalnych

## Szablony

### Szablony `xsl:template`

- Drzewo wejściowe przeglądane jest od korzenia
- Wykonywane jest dopasowywanie wzorców `match` i przetwarzane są wszystkie dopasowane elementy
- Szablony w budowane:
  - inicjacja przetwarzania korzenia drzewa
  - przepisywanie tekstowej zawartości liści drzewa na wyjście

### Przetwarzanie rekurencyjne

- Instrukcja `xsl:apply-templates` wywołuje przetwarzanie rekurencyjne

### Konflikty wzorców

- Gdy do elementu pasują wzorce kilku szablonów
- Decyduje:
  - priorytet wzorca
  - kolejność w skrypcie (ostatni najważniejszy)

### Kontekst

- Wzorce podają w spójrzędne dopasowywanych elementów określone względem bieżącego kontekstu
- Kontekstem instrukcji w ciele szablonu jest element przetwarzany przez ten szablon
- Niektóre rodzaje przetwarzania powodują zmianę bieżącego kontekstu, np.
  - uaktywnienie szablonu przez dopasowanie wzorca
  - instrukcja `xsl:for-each`
- Niektóre rodzaje przetwarzania nie powodują zmiany bieżącego kontekstu, np.
  - instrukcje warunkowe
  - wywołanie szablonu po nazwie

## Inne elementy przetwarzania

### Przetwarzanie proceduralne

- Uzupełnia przetwarzanie rekurencyjne
- Główne środki
  - instrukcja `xsl:for-each`
  - instrukcje warunkowe `xsl:if`
  - `xsl:choose`
  - wywołanie szablonów po nazwie (z przekazywaniem parametrów)

### Sortowanie

- Potrzebne w instrukcjach przetwarzających zbiory elementów, np.
  - `xsl:apply-templates`
  - `xsl:for-each`
- Definiowane osobną instrukcją `xsl:sort`, zawartą w ciele instrukcji przetwarzającej

### Tworzenie drzewa wynikowego

- Instrukcje i teksty zawarte w ciele szablonu służą do wytworzenia drzewa wynikowego
- Instrukcje służące do tworzenia struktury wynikowego dokumentu
  - `xsl:element`
  - `xsl:attribute`
  - `xsl:text`
  - `xsl:processing-instruction`
  - `xsl:comment`
- Elementy „literalne”
  - nie należące do przestrzeni nazw `xsl` ani innej znanej procesorowi XSL
  - kopiowane „literalnie” do drzewa wynikowego
  - jeśli zawierają znaczniki, to muszą być *well-formed*
- Przeniesienie zawartości elementów lub atrybutów: `xsl:value-of`
- Kopowanie fragmentów struktury: `xsl:copy`

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Formatowanie

### ➤ Zastosowania XSL

### ➤ Podsumowanie

## XPath

### Po co język XPath

- Definiuje sposób tworzenia wzorców służących do wyszukiwania elementów w dokumencie
- Używany
  - do definiowania wzorców w XSL
  - do adresowania fragmentów dokumentu w XPath
  - do wyszukiwania danych w projekcie XML Query

### Zasada działania

- Wyszukiwanie w drzewie wejściowym
- Węzły drzewa odpowiadają
  - elementom
  - atrybutom
  - instrukcjom przetwarzania
  - komentarzom
- Ścieżka określa
  - miejsce elementu w drzewie
  - warunki, które element ma spełniać
- W wyniku wyszukiwania zwracany jest zbiór wszystkich pasujących węzłów

### Budowa ścieżki

- Ścieżka „unikopodobna”
- Składa się z kroków oddzielonych ukośnikami
- Krok może być
  - spacyfikatorem współrzędnych
  - testem węzła (ew. z predykatem)

## XPath, c.d.

### Współrzędne (axis)

- Określają miejsce w drzewie wejściowym
  - względem bieżącego kontekstu
  - bezwzględne (ścieżka zaczyna się od '/')
- Zapis: specyfikator w współrzędnych, np.
  - / — korzeń drzewa
  - . — bieżący węzeł
  - .. — węzeł bezpośrednio nadrzędny
  - // — węzły podrzędne wraz z bieżącym
  - @ — atrybut

### Testy węzłów

- Podają, które elementy w określonych współrzędnych mają być wyszukane
- Zapis
  - nazwa elementu lub atrybutu
  - symbol typu węzła, np.
    - \* — dowolny element
    - @\* — dowolny atrybut
    - text() — węzeł tekstowy
  - alternatywa

### Predykaty

- Uzupełniają test węzła
- Podają warunki, które muszą spełniać szukane węzły
- Zawierają
  - porównania
  - operacje logiczne
  - operacje arytmetyczne
  - konwersje
  - operacje na tekstach
  - testy numeru węzła
- Przykłady
  - `przedmiot|last|1`
  - `przedmiot|position() mod 2 = 0`
  - `przedmiot|not|slovo_kluczowe`
  - `przedmiot|@id='KBD2'`
  - `przedmiot|//czesc_konzeptu/@id='Tresc'`
  - `przedmiot|contains(@id,'2')|slovo_kluczowe|.='Oracle'`

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Formatowanie

### ➤ Zastosowania XSL

### ➤ Podsumowanie

## XSLT

### Co to jest?

- Specyfikacja części XSL służącej do transformacji
- Znaczenie zmienione w stosunku do początkowej propozycji specyfikacji XSL

### Zmienne

- Definiowanie
  - instrukcja `xsl:variable` definiuje zmienną i nadaje jej wartość
  - zmienne zdefiniowane w elemencie `xsl:stylesheet` są globalne
- Użycie
  - używane tam, gdzie oczekiwano wyrażenia
  - nazwa jest poprzedzana znakiem '\$'

### Tryby

- Służą do rozróżnienia sposobu przetwarzania tego samego elementu w różnych kontekstach
- Można zdefiniować kilka szablonów o tej samej nazwie, różniących się atrybutem `mode`
- W instrukcji `xsl:apply-templates` można powołać się na określony tryb

### Szablony nazwane i parametry

- Szablon nazwany ma atrybut `name`
- Szablon taki można wywołać instrukcją `xsl:call-template`
- Kontekst przy takim wywołaniu nie zmienia się
- Można przekazywać parametry
  - parametry formalne definiuje się instrukcją `xsl:param` we wnętrzu szablonu
  - parametry aktualne przekazuje się instrukcją `xsl:with-param` we wnętrzu instrukcji `xsl:call-template`
  - do wartości odwołuje się przez podanie nazwy poprzedzonej znakiem '\$'

### Włączenie i import plików

- `xsl:include` włącza pliki XSL w miejsce wystąpienia instrukcji
- `xsl:import` włącza pliki XSL, ale szablony w nich zawarte mają priorytet niższy od tych z pliku głównego („biblioteki” szablonów)

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Formatowanie

### ➤ Zastosowania XSL

### ➤ Podsumowanie

## Formatowanie

### Obiekty formatujące

- Służą do formatowania dokumentów
- Używane mają być do definiowania arkuszy stylizacyjnych
- Dokument ma być przekształcany w drzewo FO
- Przeglądarka ma interpretować to drzewo i wyświetlać je w postaci odpowiednio sformatowanej
- Możliwości formatowania: ± suma HTML i CSS
- Zdefiniowano słownik obiektów formatujących i przestrzeń nazw FO
- Specyfikacja nie jest ukończona
- Nie ma implementacji w popularnych przeglądarkach

### Model prezentacji

- Budowa dokumentu
  - *page-sequence* — część dokumentu, różniąca się od innych rozkładem strony
  - *flow* — sekcja (rozdział, podrozdział itp.)
  - *block* — akapit lub podobna część
  - *inline* — część akapitu, różniąca się od innych np. ccionką
  - *wrapper* — „niewidzialny” obiekt, będący częścią bloku lub *inline*, służący do „zaczepienia” dających się dziedziczyć właściwości
  - *graphic* — referencja do zewnętrznego obiektu graficznego
  - *list* — lista
  - *table* — tabela
- Określono także złożony model stronicowania

## Formatowanie, c.d.

### Właściwości i zawartość obiektów FO

- Nazwa obiektu i jego właściwości (atrybuty) określają sposób prezentacji
- Typowe właściwości
  - ccionka
  - margines, wcięcie, odstępy
  - justowanie
- Właściwości mają wartości domyślne
- Niektóre właściwości podlegają dziedziczeniu w hierarchii podległości obiektów
- Tekst formatowany stanowi zawartość obiektu (elementu)

### Transformacje na HTML

- Powszednie stosowane do formatowania dokumentów XML
- Przyczyna — brak implementacji FO
- Możliwości użycia
  - XSLT i przetwarzanie przez odpowiedni procesor
  - stara wersja XSL i bezpośrednia prezentacja w MSIE 5

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Formatowanie

### ➤ Zastosowania XSL

### ➤ Podsumowanie

## Implementacje XSL

### Procesory XSL

- Liczne procesory XSLT związane z parserami XML, m.in. w Oracle8i

### Przeglądarki

- Procesor XSL wbudowany w MSIE 5
  - nie „rozumie” XSLT, lecz „stara” wersję XSL,
  - można doinstalować wersję β procesora XSLT, ale wywołanie wyłącznie programowe

## XSL a Oracle

### Oracle XML Developer Kit

- Współpraca z Oracle8i
- Zawiera parser XML dla języków Java, C, C++, PL/SQL
- Parserze wyposażono w procesor XSLT
- API parsera (Java) udostępnia klasy do przetwarzania XSLT
  - XSLStyleSheet
  - XSLProcessor

### Zastosowania procesora XSLT

- Współpraca z Oracle XML SQL Utility for Java
  - formatowanie wyników działania
  - przekształcanie danych wejściowych
- Współpraca z Oracle XSQL Servlet
  - formatowanie wyników działania

### Inne produkty Oracle związane z XSL

- *XML Transformer Java Beans*
  - *XSL Transformer Bean* służy do transformacji dokumentów z użyciem XSL
  - można go użyć po stronie serwera lub klienta
- *Oracle Portal-To-Go*
  - transformuje dynamicznie zawartość stron internetowych na specyficzne formaty zrozumiałe dla różnych urządzeń, np. na WML
  - informacja wejściowa jest przekształcana do generycznego formatu w XML
  - procesor XSL przekształca ją w postać wyjściową

## Przykłady zastosowań XSL

### Przetwarzanie stron WWW

- Arkusze stylizacyjne
  - przetwarzanie w przeglądarce
  - przetwarzanie na serwerach WWW
- Dynamiczne kreowanie stron
  - przetwarzanie danych kreowanych dynamicznie z zapytań np. SQL
  - dynamiczne generowanie arkuszy stylizacyjnych na podstawie preferencji użytkownika — personalizacja

### Wymiana danych

- XML jest bardzo dobrym medium do wymiany danych między bazami
- Dokument XML zawiera metainformację
- Jeśli modele baz różniły się, XSL może służyć do transformacji

### Tworzenie dokumentacji

- Do tworzenia tekstowej dokumentacji projektowej
- Sposób wykorzystania
  - dane projektowe w specjalnie zaprojektowanych dokumentach XML
  - przetwarzanie XSL do postaci sprawozdań, tabel, zestawień itp.
  - konwersja na postać docelową sprawozdania, np. RTF, TeX itp.
  - konwersja na strony HTML z łącznikami
- Zalety
  - sformalizowany zapis
  - unikanie powtarzania informacji
  - duże możliwości przetwarzania
- Szczególnie przydatne we wstępnym fazach projektów
  - elastyczność — tworzenie własnych modeli
  - darmowe narzędzia
- Przykłady
  - analiza procesów biznesowych dla *data mining*
  - sformalizowany zapis wymagań w projekcie portalu internetowego
  - projekt komunikatu typu EDI w XML

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Formatowanie

### ➤ Zastosowania XSL

### ➤ Podsumowanie

## Części i specyfikacje XSL

### Części

- XSLT (*XSL Transformations*)
  - przekształcenia drzew
- XPath — ścieżki
  - do zapisu wzorców w szablonach XSL
  - używane także w XPath'er i XML Query
- Słownik obiektów formatujących

### Specyfikacje

- Opracowane *World Wide Web Consortium (W3C)*
- XSL 1.0 — *W3C Working Draft*
- XSLT 1.0 — *W3C Recommendation*
- XPath 1.0 — *W3C Recommendation*
- Powiązanie arkuszy stylizacyjnych z dokumentami XML — *W3C Recommendation*

## Podsumowanie

- XSL jest ważnym składnikiem środowiska XML
- XSL jest dosyć silnym językiem, pozwalającym na złożone przekształcenia i zaawansowane formatowanie
- XSLT daje duże możliwości transformacji dokumentów  
XML → XML, XML → HTML, HTML → HTML itp.
- Po ukończeniu specyfikacji i zaimplementowaniu obiektów formatujących XSL uzyska duże możliwości formatowania dokumentów
- XSL jest szczególnie przydatny do tworzenia dynamicznych stron WWW
- XSL stanowi ważny element systemów wykorzystujących XML i bazy danych

## Język XSL

### ➤ Co to jest XSL

### ➤ Podstawy XSL

### ➤ XPath

### ➤ XSLT

### ➤ Zastosowania XSL

### ➤ Podsumowanie