XSL (XSLT)

Transformacja dokumentów XML.

XML – warstwa przechowująca informacje

XSL – warstwa przekształcania (**XSLT**) oraz prezentacji informacji (**XSL FO**).

Transformacja XSLT – utworzenie nowego dokumentu wynikowego na podstawie danych zawartych w pliku XML oraz pliku transformacji – nowy dokument zawiera dane oraz opis sposobu ich prezentacji.

```
<?xml version="1.0" encoding="ISO-8859-2"?>

<klient>
   <imie>Karol</imie>
   <nazwisko>Jonakowski</nazwisko>
   <data_urodzenia>1978-10-30</data_urodzenia>
   <pesel>78092002511</pesel>
   <wiek>20</wiek>
</klient>
```

[Prosty dokument XML – warstwa kodu]

```
<?xml version="1.0" encoding="ISO-8859-2" ?>
- <klient>
    <imie>Karol</imie>
    <nazwisko>Jonakowski</nazwisko>
    <data_urodzenia>1978-10-30</data_urodzenia>
    <pesel>78092002511</pesel>
    <wiek>20</wiek>
  </klient>
```

[Ten sam dokument widziany w przeglądarce internetowej – bez przekształceń]

Klient	
Imie:	Karol
Nazwisko:	Jonakowski
Data urodzenia:	1978-10-30
PESEL:	78092002511
Wiek:	20

[Ponownie ten sam plik, tym razem po przekształceniu przy użyciu XSLT]

Ćwiczenie 1.

Źródła: pliki 1_e.xml | 1_e.xsl

Plik XML odpowiada za przechowywanie danych. Nie umieszczamy w nim żadnych instrukcji dotyczących sposobu prezentacji informacji. Dołączamy natomiast odpowiednie odwołanie do szablonu:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xml-stylesheet type="text/xsl" href="1_e.xsl"?>

<klient>
   <imie>Karol</imie>
   <nazwisko>Jonakowski</nazwisko>
   <data_urodzenia>1978-10-30</data_urodzenia>
   <pesel>78092002511</pesel>
   <wiek>20</wiek>
</klient>
```

[Plik 1_e.xml]

Dołączenie instrukcji przetwarzania wskazującej na arkusz stylów 1_a.xsl

Dołączony wzorzec wyświetlania:

```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">

<xsl:template match="/">

<html>
   <head>
      <title>Klient</title>
   </head>
   <body>
      <table width="200" border="1">
         <tbody>
            <tr>
               <th colspan="2" align="center">Klient</th>
            </tr>

            <tr>
               <td align="right">Imie:</td>
               <td align="center">
                  <xsl:value-of select="klient/imie"/>
               </td>
            </tr>
```

```
            <tr>
               <td align="right">Nazwisko:</td>
               <td align="center">
                  <xsl:value-of select="klient/nazwisko"/>
               </td>
            </tr>
            <tr>
               <td align="right">Data urodzenia:</td>
               <td align="center">
                  <xsl:value-of select="klient/data_urodzenia"/>
               </td>
            </tr>
            <tr>
               <td align="right">PESEL:</td>
               <td align="center">
                  <xsl:value-of select="klient/pesel"/>
               </td>
            </tr>
            <tr>
               <td align="right">Wiek:</td>
               <td align="center">
                  <xsl:value-of select="klient/wiek"/>
               </td>
            </tr>
         </tbody>
      </table>
   </body>
</html>

</xsl:template>

</xsl:stylesheet>
```

[Plik 1_e.xsl]

<xsl:template match="...">

instrukcja określa szablon dla elementu podanego jako wartość atrybutu match

<xsl:value-of select="...">

instrukcja pobiera i wypisuje zawartość elementu podanego jako wartość atrybutu select

Klient	
Imie:	Karol
Nazwisko:	Jonakowski
Data urodzenia:	1978-10-30
PESEL:	78092002511
Wiek:	20

[dane z pliku 1_e.xml wyświetlone zgodnie z szablonem 1_e.xsl]

Ćwiczenie 2

Źródła: pliki 1_a.xml | 1_a.xsl

Dane w plikach XML możemy zapisać jako wartości elementów lub dla atrybutów. W poprzednim ćwiczeniu użyliśmy pliku XML, który wykorzystywał tylko elementy. Tym razem użyjemy także atrybutów.

Plik 1_a.xml jest zmodyfikowanym dokumentem z poprzedniego ćwiczenia. Do elementu klient został tu dodany atrybut typ:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xml-stylesheet type="text/xsl" href="1_a.xsl"?>
<klient typ="Odbiorca">
   <imie>Karol</imie>
   <nazwisko>Jonakowski</nazwisko>
   <data_urodzenia>1978-10-30</data_urodzenia>
   <pesel>78092002511</pesel>
   <wiek>20</wiek>
</klient>
```

[plik 1_a.xml]

Odpowiednim zmianom został poddany także szablon prezentacji:

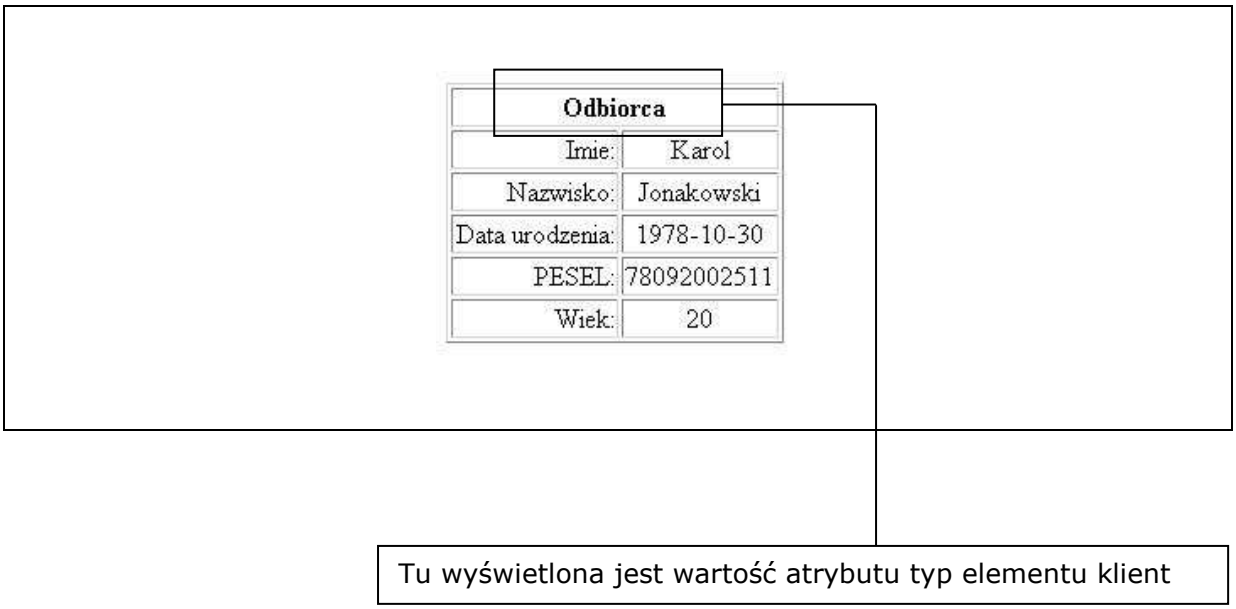
```
<?xml version="1.0" encoding="UTF-8"?>

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<xsl:template match="/">

<html>
   <head>
      <title>Klient</title>
   </head>
   <body>
      <table width="200" border="1">
         <tbody>
            <tr>
               <th colspan="2" align="center">
                  <xsl:value-of select="klient/@typ"/>
               </th>
            </tr>
            <tr>
               <td align="right">Imie:</td>
               <td align="center"><xsl:value-of select="klient/imie"/></td>
            </tr>
            <tr>
               <td align="right">Nazwisko:</td>
               <td align="center"><xsl:value-of select="klient/nazwisko"/></td>
            </tr>
```

```
            <tr>
               <td align="right">Data urodzenia:</td>
               <td align="center"><xsl:value-of select="klient/data_urodzenia"/></td>
            </tr>
            <tr>
               <td align="right">PESEL:</td>
               <td align="center"><xsl:value-of select="klient/pesel"/></td>
            </tr>
            <tr>
               <td align="right">Wiek:</td>
               <td align="center"><xsl:value-of select="klient/wiek"/></td>
            </tr>
         </tbody>
      </table>
   </body>
</html>

</xsl:template>

</xsl:stylesheet>
```

<xsl:value-of select="../@..">

wartość atrybutu pobieramy analogicznie jak wartość elementu

nazwę atrybutu poprzedzamy @



Ćwiczenie 3

Źródła: pliki 2_a.xml | 2_a.xsl

Często zdarza się iż w pliku XML mamy wiele danych w tym samym formacie. Najczęściej są to struktury zbliżone do listy. Takie dane zawiera poniższy plik:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xml-stylesheet type="text/xsl" href="2_a.xsl"?>
<klienci>
   <klient typ="Odbiorca">
      <imie>Karol</imie>
      <nazwisko>Jonakowski</nazwisko>
      <data_urodzenia>1978-10-30</data_urodzenia>
      <pesel>78092002511</pesel>
      <wiek>25</wiek>
   </klient>
   <klient typ="Dostawca">
      <imie>Marian</imie>
      <nazwisko>Kaczmarek</nazwisko>
      <data_urodzenia>1981-11-10</data_urodzenia>
      <pesel>81111003521</pesel>
      <wiek>22</wiek>
   </klient>
   <klient typ="Odbiorca">
      <imie>Adam</imie>
      <nazwisko>Miernik</nazwisko>
      <data_urodzenia>1967-09-12</data_urodzenia>
      <pesel>67091203521</pesel>
      <wiek>37</wiek>
   </klient>
</klienci>
```

[2_a.xml]

Mamy tu do czynienia z listą klientów. Część z nich to nasi odbiorcy, część to dostawcy. Ze względu na analogie pomiędzy obiema kategoriami, do przechowania informacji użyto jednego pliku.

Dołączamy do niego odpowiedni szablon:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">

<html>

   <head>
      <title>Klient</title>
   </head>
```

```
   <body>
      <xsl:for-each select="klienci/klient">
         <table width="200" border="1">
            <tbody>
               <tr>
                  <th colspan="2" align="center"><xsl:value-of select="@typ"/></th>
               </tr>
               <tr>
                  <td align="right">Imie:</td>
                  <td align="center"><xsl:value-of select="imie"/></td>
               </tr>
               <tr>
                  <td align="right">Nazwisko:</td>
                  <td align="center"><xsl:value-of select="nazwisko"/></td>
               </tr>
               <tr>
                  <td align="right">Data urodzenia:</td>
                  <td align="center"><xsl:value-of select="data_urodzenia"/></td>
               </tr>
               <tr>
                  <td align="right">PESEL:</td>
                  <td align="center"><xsl:value-of select="pesel"/></td>
               </tr>
               <tr>
                  <td align="right">Wiek:</td>
                  <td align="center"><xsl:value-of select="wiek"/></td>
               </tr>
            </tbody>
         </table>
         <br/>
      </xsl:for-each>
   </body>

</html>

</xsl:template>

</xsl:stylesheet>
```

[2_a.xsl]

<xsl:for-each select="...">

instrukcje wykonane zostaną dla każdego elementu pasującego do wartości określonej w jako wartość atrybutu select

Efekt działania szablonu 2_a.xsl:

Odbiorca	
Imie:	Karol
Nazwisko:	Jonakowski
Data urodzenia:	1978-10-30
PESEL:	78092002511
Wiek:	25

Dostawca	
Imie:	Marian
Nazwisko:	Kaczmarek
Data urodzenia:	1981-11-10
PESEL:	81111003521
Wiek:	22

Odbiorca	
Imie:	Adam
Nazwisko:	Miernik
Data urodzenia:	1967-09-12
PESEL:	67091203521
Wiek:	37

Dla każdego klienta wygenerowana została oddzielna tabela.

Ćwiczenie 4

Źródła: **plik 2_a.xml**

Należy utworzyć szablon XSL, prezentujący dane zawarte w pliku 2_a.xml w następującej postaci:

Imie	Nazwisko	Data Urodzenia	PESEL	Wiek	Typ
Marian	Kaczmarek	1981-11-10	81111003521	22	Dostawca
Karol	Jonakowski	1978-10-30	78092002511	25	Odbiorca
Adam	Miernik	1967-09-12	67091203521	37	Odbiorca