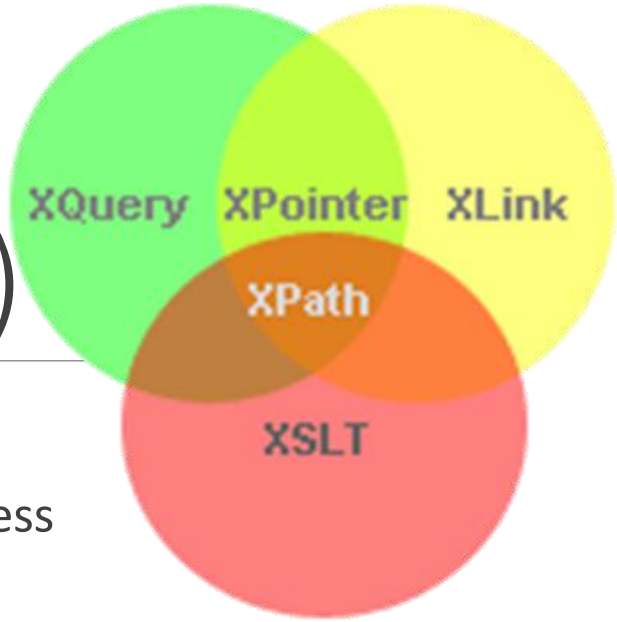
XSL

DR WIOLETA SZWOCH

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

XPath (*XML Path Language*)



standard for identifying parts of an XML document

An unambiguous description of the element's address in the XML file

Used in other standards, does not exist alone

Path expressions are used to navigate in XML documents

- syntax similar to Unix file system paths
- the ability to extract the necessary nodes

XPath contains a library of standard functions

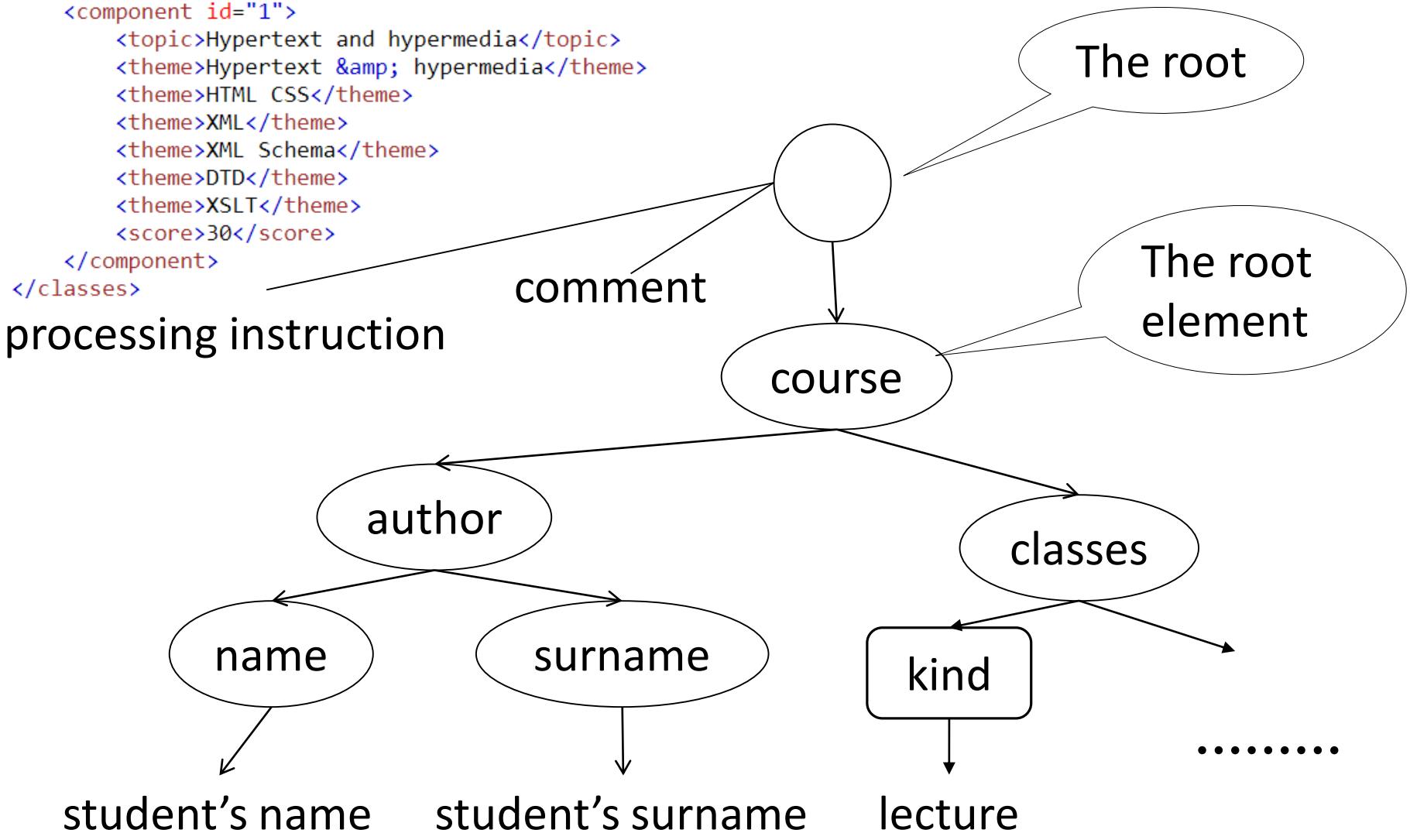
Nodes

XML document

- Tree structure with nodes
- Each node represents part of XML document
  - Seven types
    - Root
    - Element
    - Attribute
    - Text
    - Comment
    - Processing instruction
    - Namespace

Data model for XPath

```
<course>
    <author>
        <name>student's name</name>
        <surname>student's surname</surname>
    </author>
    <classes kind="lecture" >
        <component id="1">
            <topic>Hypertext and hypermedia</topic>
            <theme>Hypertext &amp;amp; hypermedia</theme>
            <theme>HTML CSS</theme>
            <theme>XML</theme>
            <theme>XML Schema</theme>
            <theme>DTD</theme>
            <theme>XSLT</theme>
            <score>30</score>
        </component>
    </classes>
```



```xml
<course>
    <author>
        <name>student's name</name>
        <surname>student's surname</surname>
    </author>
    <classes kind="lecture" >
        <component id="1">
            <topic>Hypertext and hypermedia</topic>
            <theme>Hypertext &amp;amp; hypermedia</theme>
            <theme>HTML CSS</theme>
            <theme>XML</theme>
            <theme>XML Schema</theme>
            <theme>DTD</theme>
            <theme>XSLT</theme>
            <score>30</score>
        </component>
    </classes>
```

XPath is a syntax for "addressing" into a document.
They are "path expressions".
XPath expressions have a directory-path-like syntax.

relative path
step/step/...
absolute path
/step/step/...
A step consists of:
an axis
a node-test
zero or more predicates

Step – full syntax:
axis::**node-test** [**predicate1**] [**predicate2**] ...
axis – direction in document tree
node-test – selecting nodes by kind, name, or type
predicates – (0 or more) additional logical conditions for filtering

```xml
<course>
    <author>
        <name>student's name</name>
        <surname>student's surname</surname>
    </author>
    <classes kind="lecture" >
        <component id="1">
            <topic>Hypertext and hypermedia</topic>
            <theme>Hypertext &amp;amp; hypermedia</theme>
            <theme>HTML CSS</theme>
            <theme>XML</theme>
            <theme>XML Schema</theme>
            <theme>DTD</theme>
            <theme>XSLT</theme>
            <score>30</score>
        </component>
    </classes>
```

XPath is a syntax for "addressing" into a document.
They are "path expressions".
XPath expressions have a directory-path-like syntax.

"**/**" represents the Document info item (root)

* matches any element author/*

@ means attribute classes/@kind

classes//theme //theme

The result of evaluating an XPath expression is a Node Set

"//" matches elements that aren't direct children

XPath

XPath treat XML as a tree of elements

Root of tree – document node main element (aka document element) is not the root

„Leaf” may be:

- tag, attribute, processing instruction, comment, text, namespace

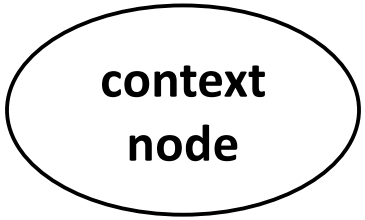
”leaves” are related by ”branches” - axes

Nodes

Relations between nodes

- child, parent, descendant , ancestor, sibling

```
<course>
    <author>
        <name>student's name</name>
        <surname>student's surname</surname>
    </author>
    <classes kind="lecture" >
        <component id="1">
            <topic>Hypertext and hypermedia</topic>
            <theme>Hypertext &amp;amp; hypermedia</theme>
            <theme>HTML CSS</theme>
            <theme>XML</theme>
            <theme>XML Schema</theme>
            <theme>DTD</theme>
            <theme>XSLT</theme>
            <score>30</score>
        </component>
    </classes>
```



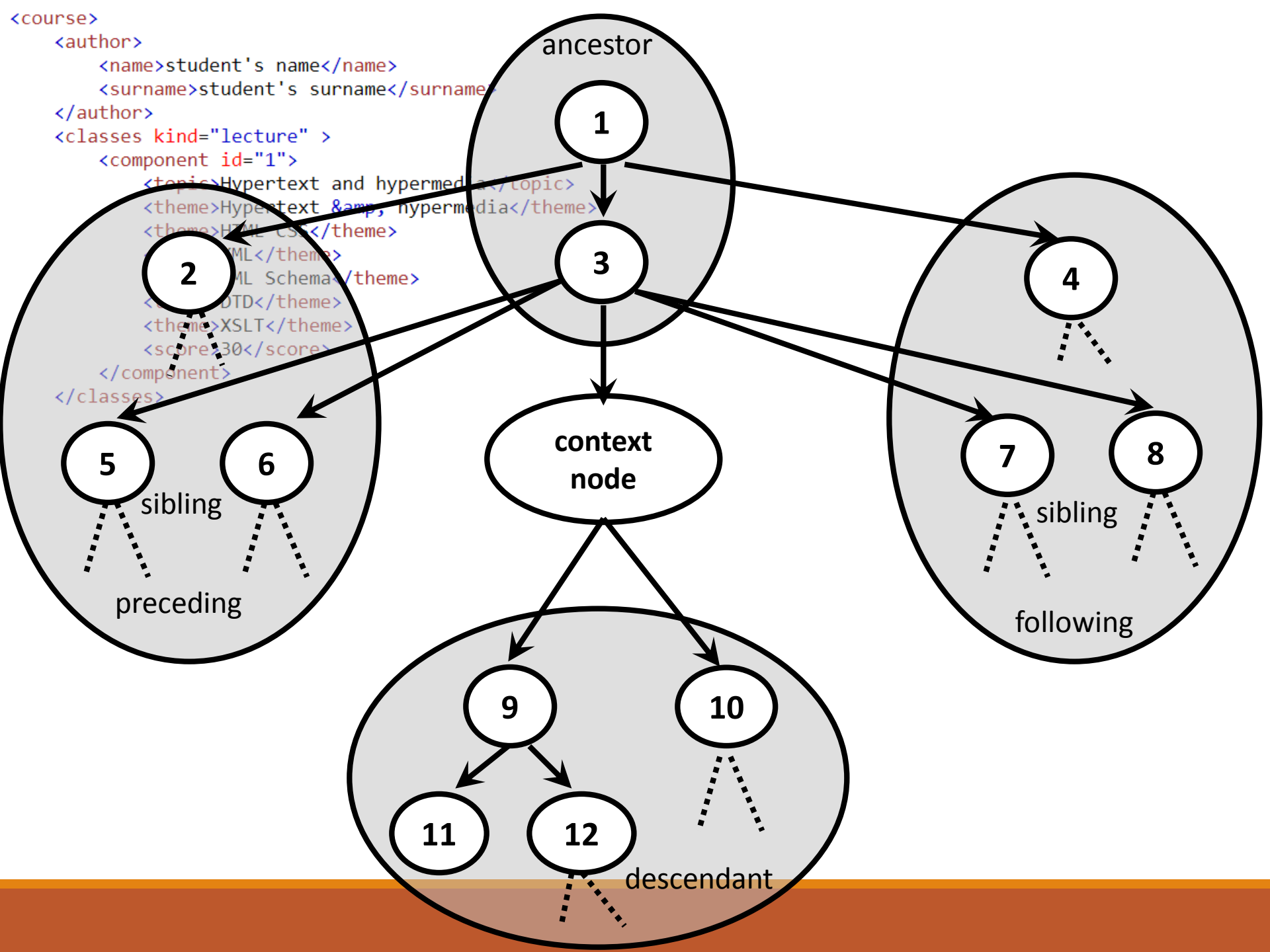
You can address the context node as '**.**'

./@*

The context node is implicit.

range/component ≡ ./range/component

The context node does not have to be an element.


<course>
<author>
<name>student's name</name>
<surname>student's surname</surname>
</author>
<classes kind="lecture" >
<component id="1">
<topic>Hypertext and hypermedia</topic>
<theme>Hypertext & hypermedia</theme>
<theme>HTML CSS</theme>
ML</theme>
ML Schema</theme>
DTD</theme>
<theme>XSLT</theme>
<score>30</score>
</component>
</classes>
ancestor
1
2
3
4
5
6
sibling
preceding
context node
7
8
sibling
following
9
10
11
12
descendant

Axis

self
child
descendant
parent
ancestor
following-sibling
preceding-sibling
following
preceding
attribute
namespace
descendant-or-self
ancestor-or-self

Node test
by kind of node
node()
text()
comment()
processing-instruction()
attribute()
by name

.//classes/@kind
expands to
self::**node()**/
descendant-or-self::**node()**/ **child**::classes/
attribute::kind

Built-in XPath functions and operators

Text
- concat(s1, s2, ...) substring(s, pos, len)
  starts-with(s1, s2) contains(s1, s2) string-length(s) translate(s, t1, t2)

Numbers
- floor(x) ceiling(x) round(x)

Nodes
- name(n?) local-name(n?) namespace-uri(n?)

Sequences
- count(S) sum(S) min(S) max(S) avg(S) empty(S) reverse(S) distinct-values(S)

Context
- current() position() last()

Arithmetic - * div mod

Logical values and or

Comparison operators = != < <= > >=

Predicates

additional logical conditions for filtering

/path[predicate]

- they allow you to check properties that can not be expressed in the nodes themselves
- any XPath expression
- predicates may contain functions and operators
- only nodes for which the predicate is true are included in the result

```
<course>
    <author>
        <name>student's name</name>
        <surname>student's surname</surname>
    </author>
    <classes kind="lecture" >
        <component id="1">
            <topic>Hypertext and hypermedia</topic>
            <theme>Hypertext &amp; hypermedia</theme>
            <theme>HTML CSS</theme>
            <theme>XML</theme>
            <theme>XML Schema</theme>
            <theme>DTD</theme>
            <theme>XSLT</theme>
            <score>30</score>
        </component>
    </classes>
```

component/theme[last()]
component/theme[position()<3]
/course/author/name
author | classes/component/topic
theme[3]
classes[@kind='lecture']

XSL (*eXtensible Stylesheet Language*)

SGML(1986) ($\rightarrow$ XML)

DSSSL ($\rightarrow$XSL)

- *Document Style and Semantics Specification Language*
- language for processing and transforming SGML documents into a form that can be displayed or printed
- ***A HUGE MONSTER OF A LANGUAGE***

XSL (*eXtensible Stylesheet Language*)

XSLT *(XSL Transformation)*

XSL FO *(XSL Formating Objects)*

A language that allows you to transform and display data from XML documents
What would have happened if there hadn't been an XSLT

XML document without XSL

```
<?xml version="1.0" encoding="WINDOWS-1250"?>
<?xml-stylesheet type="text/xsl" href="podroz.xslt"?>
<podroz xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="podroz.xsd">
        <etap id="1">
                <państwo>
                        <nazwa_państwa>
                        Egipt
                        </nazwa_państwa>
                        <nazwa_obca>
                        Dżumhurijat Misr al-Arabijja
                        </nazwa_obca>
                        <mieszkańcy jednostka_m="milionów">68.360</mieszkańcy>
                        <powierzchnia jednostka="km_2">1000449</powierzchnia>
                        <języki>
                                <język_urzędowy>  arabski </język_urzędowy>
                                <inne>
                                <język_inny> angielski  </język_inny>
                                </inne>
                                <inne>
                                <język_inny>  francuski  </język_inny>
                                </inne>
                        </języki>
                </państwo>
                <kiedy>
                        <data_przyjazdu>2004-07-02</data_przyjazdu>
                        <data_odjazdu>2004-07-11</data_odjazdu>
                </kiedy>
                <miasto>
                        <zdjęcie src="hurghada.jpg" width="200" height="200"/>
                        <nazwa_miasta>Hurghada</nazwa_miasta>
                        <opis>
                                <położenie>
                                Hurghada leży nad  lazurowym, kryształowo czystym Morzem Czerwonym obfitującym w bajecznie
                                kolorowe podwodne łąki raf koralowych  w odległości 500 km od  Kairu i 270 km od Luksoru. Dzięki
                                wspaniałemu usytuowaniu jest obecnie centrum wakacyjnego wypoczynku w  Egipcie. Miasto rozciąga się
                                na ok. 35 km długości, na której w większości znajdują się luksusowe hotele.
                                </położenie>
                                <historia>
                                kurort jest sławny dopiero od kilkunastu lat. Aż trudno w to uwierzyć, ale dopiero
                                od lat siedemdziesiątych ta spokojna osada rybacka jest rozbudowywana i liczy obecnie  mieszkańców.
                                </historia>
                                <klimat>
                                Dominuje klimat zwrotnikowy, pustynny
                                prawie z całkowitym brakiem opadów i
                                znacznymi wahaniami temperatur
                                pomiędzy dniem i nocą. Występujące tutaj
                                upały na wybrzeżu łagodzone są stałą
                                bryzą od Morza Czerwonego.
                                </klimat>
                                <atrakcje>
                                        <co_robić>
                                        Hurghada jest też idealnym miejscem dla osób kochających windsurfing, gdyż silnych wiatrów i
sprzyjających fal nigdy tu nie brakuje. Przyjazny klimat daje możliwość wypoczynku i pełnego relaksu przez cały rok. Jest to
także doskonałe miejsce dla tych, którzy chcą poczuć atmosferę arabskiego miasta. Wiesiamowite wrażenie mogą pozostawić zakupy
na tradycyjnym egipskim targu - souku, który przesycony jest egzotyką i orientalnym klimatem. Tutaj można zaopatrzyć się w
rozmaite pamiątki, które zawsze będą przypominały o wakacjach w Egipcie. Warto kupić papirusy, wyroby z miedzi, skóry, płetną
biżuterię ze złota i srebra, szkło, orientalne przyprawy, ręcznie tkane dywany i wiele ch wyrobów. Targowanie jest tam ogólnie
przyjętą formą kupowania. Żeby uzyskać korzystną cenę trzeba jednak poświęcić trochę czasu. Warto zobaczyć także Akwarium
położone w centrum miasta i rzymskie ruiny w Gebel Abu Dukhan. Podczas pobytu w Hurghadzie można skorzystać z rejsu na płetną
wyspę Giftun, która oddalona jest o godzinę drogi od wybrzeży Morza Czerwonego. Wyspa ta jest całkowicie pozbawiona
roślinności. Jedyne, co można tam zastać to muszle i plaseк. Jest to jednak idealna baza wypadowa dla wszystkich miłośników
nurkowania
                                        </co_robić>
                                        <cie>ciekawe_miejsca>Giftun - wyspa koralowa</cie>
                                        <cie><ciekawe_miejsca>Aquascope - możliwość podziwiania podwodnego życia przez dno łodzi ze
szklanym dnem, która zanurza się  na głębokość4 metrów.</ciekawe_miejsca></cie>
```

XML document with XSL

Krótkie informacje o państwie :

Data przyjazdu: 2004-07-02
Data odjazdu: 2004-07-11

Nazwa państwa :	Egipt
Nazwa państwa w języku urzędowym :	Dżumhurijat Misr al-Arabija
Język urzędowy :	arabski
Języki inne :	• angielski • francuski
Ilość mieszkańców :	68.360miliona
Powierzchnia :	1000449km^2

Odwiedzone przeze mnie miasto to **Hurghada.**

Hurghada leży nad lazurowym, kryształowo czystym Morzem Czerwonym obfitującym w bajecznie kolorowe podwodne łąki raf koralowych w odległości 500 km od Kairu i 270 km od Luksoru. Dzięki wspaniałemu usytuowaniu jest obecnie centrum wakacyjnego wypoczynku w Egipcie. Miasto rozciąga się na ok. 35 km długości, na której w większości znajdują się luksusowe hotele.

Troszkę historii:

Kurort jest sławny dopiero od kilkunastu lat. Aż trudno w to uwierzyć, ale dopiero od lat siedemdziesiątych ta spokojna osada rybacka jest rozbudowywana i liczy obecnie mieszkańców.

Klimat:

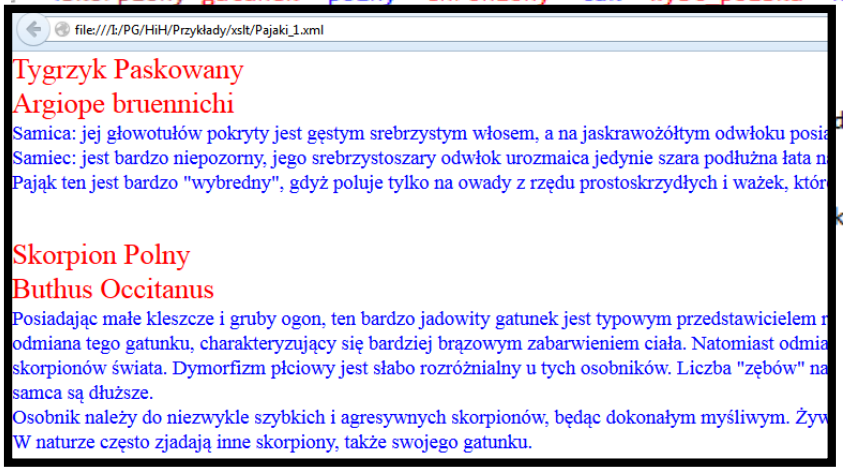
XSL (*eXtensible Stylesheet Language*)

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xml-stylesheet type="text/xsl" href="Pajeczaki2.xsl"?>
<pajeczaki xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <pajaki gatunek="tygrzyk"  chroniony="tak" wyst_polska="tak">
      <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
      <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
      <gromada>Pajeczaki</gromada>
      <wystepowanie>
        <swiat>...</swiat>
        <tereny>...</tereny>
       <srodowisko typ="lądowe" />
      </wystepowanie>
      <opis czego="Cechy">Samica: jej głowotułów pokryty jest gęstym s
      </opis>
      <opis czego="Pokarm">Pająk ten jest bardzo "wybredny", gdyż polu
      </opis>
      <opis czego="Wymiary">Samica osiąga długość ok. 25 mm. Samiec je
      </opis>
      <zdjecie zrodlo="Gfx\tygrzyk10_mlody2.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk12_ml_samica.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk12_ml_samiec.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk14_obiad.jpg"></zdjecie>
      <linki adres="http://pl.wikipedia.org/wiki/Tygrzyk_paskowany">wi
      <linki adres="http://www.salamandra.org.pl/magazyn/b08a07.html">
      <linki adres="http://www.eko.org.pl/otop-leszno/tygrzyk.php">eko
      <linki adres="http://www.bocian.org.pl/biuletyn/2001/b2a23.php">
      <linki adres="http://grzesiak.kei.pl/jurek/lista161.html">grzesi
      <data>2012-12-01</data>
  </pajaki>
```

```
<?xml version="1.0" encoding="UTF-16"?>
- <zwierzeta>
  - <tygrzyk>
      <opis>Samica: jej głowotułów pokryty jest gęst
        upodabnia ją nieco do osy. Samiec: jest bar
        bokach. </opis>
      <zdjecie>Gfx\tygrzyk10_mlody2.jpg</zdjecie>
    </tygrzyk>
  - <krzyzak>
      <opis>Na odwłoku jasne plamy układają się w c
        odcień brązu. Cechą charakterystyczną jest
        pozbawiony nóg, przeważnie jest pękaty. Sa
      <zdjecie>Gfx/krzyzak.jpg</zdjecie>
    </krzyzak>
  - <polny>
      <opis>Posiadając małe kleszcze i gruby ogon, t
        mniej jadowita odmiana tego gatunku, char
        należy do najbardziej jadowitych skorpionó
        od 19 do 30, natomiast u samca 25 do 36. P
      <zdjecie>Gfx/skorpion1.jpg</zdjecie>
    </polny>
</zwierzeta>
```

Displaying XML using CSS

```xml
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="p1.css" ?>
<pajeczaki>
  <pajaki gatunek="tygrzyk"  chroniony="tak" wyst_polska="tak">
    <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
    <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
    <opis czego_dotyczy="Cechy">
      Samica: jej głowotułów pokryty jest gęstym srebrzystym włose
    </opis>
    <opis czego_dotyczy="Pokarm">
      Pająk ten jest bardzo "wybredny", gdyż poluje tylko na owady
    </opis>
  </pajaki>
  <skorpiony gatunek="polny" chroniony="tak" wyst_polska="nie">
```




```
dowity g
korpionó
```

```css
pajeczaki
{
background-color: #ffffff;
width: 100%;
}
pajaki, skorpiony
{
display: block;
margin-bottom: 30pt;
margin-left: 0;
}
nazwa
{
display: block;
color: #FF0000;
font-size: 20pt;
}
opis
{
display: block;
color: #0000FF;
font-size: 14pt;
}
```

XSLT *E**X**tensible **S**tylesheet **L**anguage **T**ransformations*

CSS – Cascading Style Sheet

- style sheet determines the style or the appearance of tags (HTML or XML)
- we can define fonts, margins, colours, borders, …

XSLT *E**X**tensible **S**tylesheet **L**anguage **T**ransformations*

CSS – Cascading Style Sheet

- style sheet determines the style or the appearance of tags (HTML or XML)
- we can define fonts, margins, colours, borders, …

XSLT stylesheet

- a document that generates data based on XML document, the resulting document may or may not contain formatting information
- a complete high-level language for manipulating an XML documents
- it does not replace existing programming languages but complements them

XSLT *E**X**tensible **S**tylesheet **L**anguage **T**ransformations*

the language of transforming XML trees

the basic processing paradigm is pattern matching

declarative language

data-driven language (push processing model)

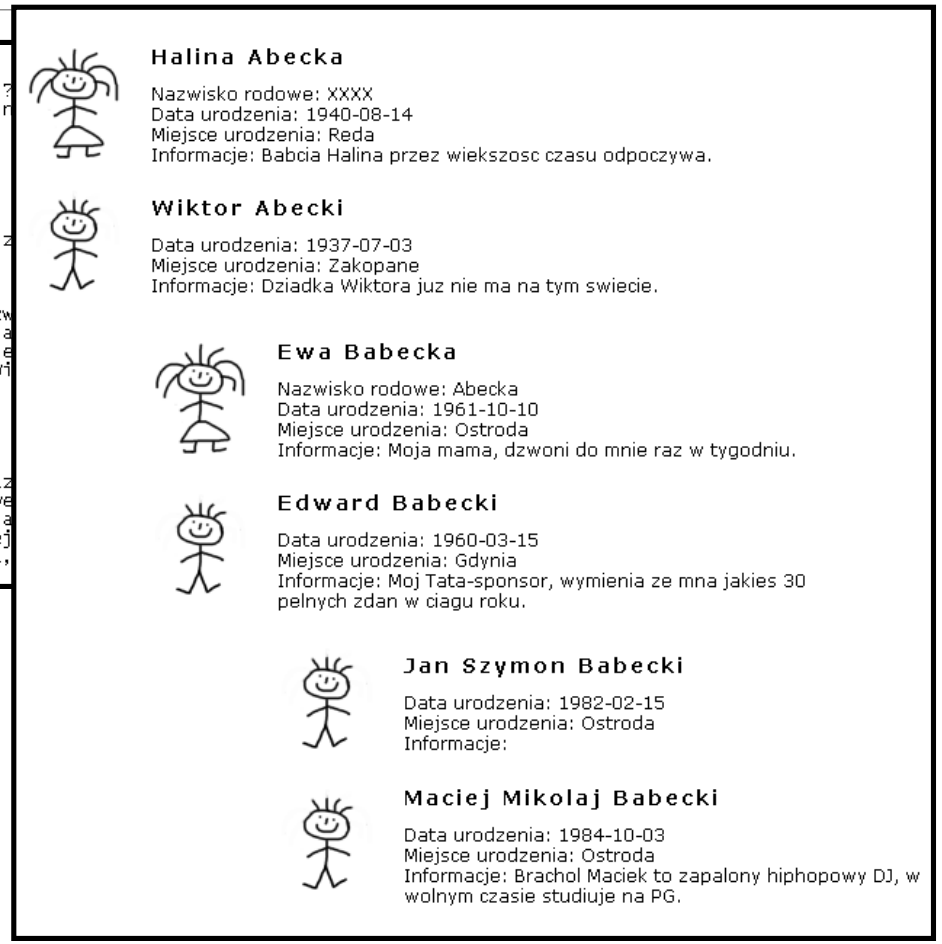
- the code is executed in response to the data;
- the code is executed **nondeterministically**

XSLT Advantages

Independent of programming.	• Transformations are written in a separate XSL file which is an XML document.
Output can be changed by simply modifying the transformations in an XSL file.	• No need to change any code. So Web designers can edit the stylesheet and can see the change in the output quickly.
allows you to work with XML documents	• we do not have to write programs to process XML
different document type from XML	• (HTML, XML, CSV, …)
different XML – different XSLT	• the same transformation can be applied to many XML documents • different XSLT can be applied to the same XML document

Example

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)-->
<?xml-stylesheet type="text/xsl" href="bk-drzewo.xsl"?
<drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-in
        <osoba plec="kobieta">
                <imie>Halina</imie>
                <nazwisko>Abecka</nazwisko>
                <rodowe>XXXX</rodowe>
                <data>1940-08-14</data>
                <miejsce>Reda</miejsce>
                <informacje>Babcia Halina przez wieksz
                <partner>
                        <osoba plec="mezczyzna">
                                <imie>Wiktor</imie>
                                <nazwisko>Abecki</nazw
                                <data>1937-07-03</data
                                <miejsce>Zakopane</mie
                                <informacje>Dziadka wi
                        </osoba>
                </partner>
                <dzieci>
obieta">
Ewa</imie>
sko>Babecka</naz
e>Abecka</rodowe
1961-10-10</data
ce>Ostroda</miej
macje>Moja mama,
er>
```



Halina Abecka

Nazwisko rodowe: XXXX
Data urodzenia: 1940-08-14
Miejsce urodzenia: Reda
Informacje: Babcia Halina przez wiekszosc czasu odpoczywa.

Wiktor Abecki

Data urodzenia: 1937-07-03
Miejsce urodzenia: Zakopane
Informacje: Dziadka Wiktora juz nie ma na tym swiecie.

Ewa Babecka

Nazwisko rodowe: Abecka
Data urodzenia: 1961-10-10
Miejsce urodzenia: Ostroda
Informacje: Moja mama, dzwoni do mnie raz w tygodniu.

Edward Babecki

Data urodzenia: 1960-03-15
Miejsce urodzenia: Gdynia
Informacje: Moj Tata-sponsor, wymienia ze mna jakies 30 pelnych zdan w ciagu roku.

Jan Szymon Babecki

Data urodzenia: 1982-02-15
Miejsce urodzenia: Ostroda
Informacje:

Maciej Mikolaj Babecki

Data urodzenia: 1984-10-03
Miejsce urodzenia: Ostroda
Informacje: Brachol Maciek to zapalony hiphopowy DJ, w wolnym czasie studiuje na PG.

Anna Nowak

Nazwisko rodowe: XXXX
Data urodzenia: 1940-08-14
Miejsce urodzenia: Reda
Informacje: Babcia

Andrzej Nowak

Data urodzenia: 1940-07-03
Miejsce urodzenia: Torun
Informacje: Dziadek

Katarzyna Lusnia

Nazwisko rodowe: Nowak
Data urodzenia: 1963-10-10
Miejsce urodzenia: Kalisz
Informacje: Mama

Jan Lusnia

Data urodzenia: 1960-03-15
Miejsce urodzenia: Gdynia
Informacje: Tata

Edward Szymon Lusnia

Data urodzenia: 1982-02-15

XSLT

XML

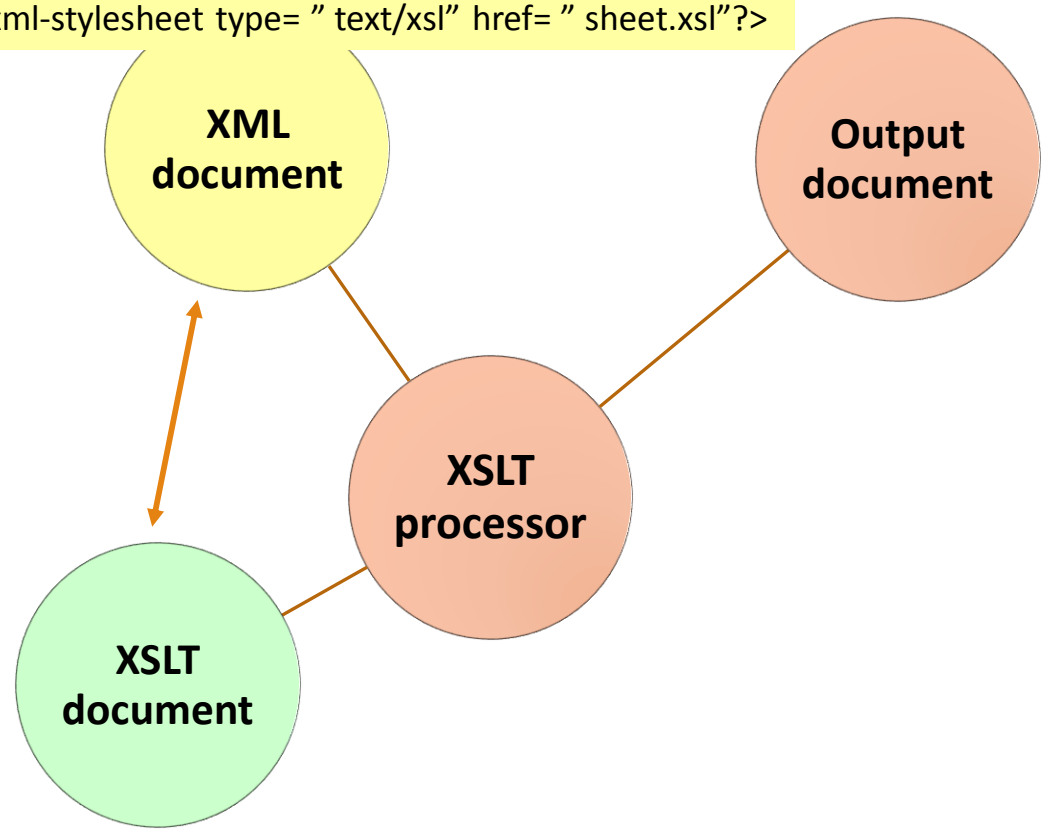
```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)-->
<?xml-stylesheet type="text/xsl" href="bk-drzewo.xsl"?>
<drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="bk-drzewo.xsd">
        <osoba plec="kobieta">
                <imie>Halina</imie>
                <nazwisko>Abecka</nazwisko>
                <rodowe>XXXX</rodowe>
                <data>1940-08-14</data>
                <miejsce>Reda</miejsce>
                <informacje>Babcia Halina przez wiekszosc czasu odpoczywa.</informacje>
                <partner>
                        <osoba plec="mezczyzna">
                                <imie>wiktor</imie>
                                <nazwisko>Abecki</nazwisko>
                                <data>1937-07-03</data>
                                <miejsce>Zakopane</miejsce>
                                <informacje>Dziadka wiktora juz nie ma na tym swiecie.</informacje>
                        </osoba>
                </partner>
                <dzieci>
                        <osoba plec="kobieta">
                                <imie>Ewa</imie>
                                <nazwisko>Babecka</nazwisko>
                                <rodowe>Abecka</rodowe>
                                <data>1961-10-10</data>
                                <miejsce>Ostroda</miejsce>
                                <informacje>Moja mama, dzwoni do mnie raz w tygodniu.</informacje>
                                <partner>
```

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)-->
<?xml-stylesheet type="text/xsl" href="bk-drzewo.xsl"?>
<drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="bk-drzewo.xsd">
        <osoba plec="kobieta">
                <imie>Anna</imie>
                <nazwisko>Nowak</nazwisko>
                <rodowe>XXXX</rodowe>
                <data>1940-08-14</data>
                <miejsce>Reda</miejsce>
                <informacje>Babcia</informacje>
                <partner>
                        <osoba plec="mezczyzna">
                                <imie>Andrzej</imie>
                                <nazwisko>Nowak</nazwisko>
                                <data>1940-07-03</data>
                                <miejsce>Torun</miejsce>
                                <informacje>Dziadek</informacje>
                        </osoba>
                </partner>
                <dzieci>
                        <osoba plec="kobieta">
                                <imie>Katarzyna</imie>
                                <nazwisko>Lusnia</nazwisko>
                                <rodowe>Nowak</rodowe>
                                <data>1963-10-10</data>
                                <miejsce>Kalisz</miejsce>
                                <informacje>Mama</informacje>
                                <partner>
                                        <osoba plec="mezczyzna">
                                                <imie>Jan</imie>
                                                <nazwisko>Lusnia</nazwisko>
                                                <data>1960-03-15</data>
                                                <miejsce>Gdynia</miejsce>
                                                <informacje>Tata</informacje>
                                        </osoba>
                                </partner>
                                <dzieci>
                                        <osoba plec="mezczyzna">
                                                <imie>Edward</imie>
                                                <drugie>Szymon</drugie>
                                                <nazwisko>Lusnia</nazwisko>
                                                <data>1982-02-15</data>
                                                <miejsce>Ostroda</miejsce>
                                                <informacje></informacje>
                                        </osoba>
                                        <osoba plec="kobieta">
                                                <imie>Hanna</imie>

                                                <nazwisko>Lusnia</nazwisko>
                                                <data>1995-05-29</data>
                                                <miejsce>Ostroda</miejsce>
                                                <informacje>siostra</informacje>
                                        </osoba>
```

XSLT

```
<?xml-stylesheet type= ” text/xsl” href= ” sheet.xsl”?>
```



Output document

- an **HTML** webpage, to publish the data on the web
- a **CSV** file, to easily import into a database/spreadsheet
- a **PDF** file for printing
- another **XML** file, with different tag names or layout
- other **text** files

Stylesheet processing

XSLT processor gets an XML document and a stylesheet. It starts a traversal at the **root node** and checks to see if there is a **template** match

If so, it applies the template and looks for an "xsl:apply-templates" element

If not, it traverses down the tree looking for a template match with some other node of the tree

If such an element exists, it continues the traversal

if no such element exists, the traversal stops

Structure of the XSLT document

XML declaration

xsl:stylesheet tag

- xmlns:xsl attribute: URI for XSLT namespace
  - xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
- Version

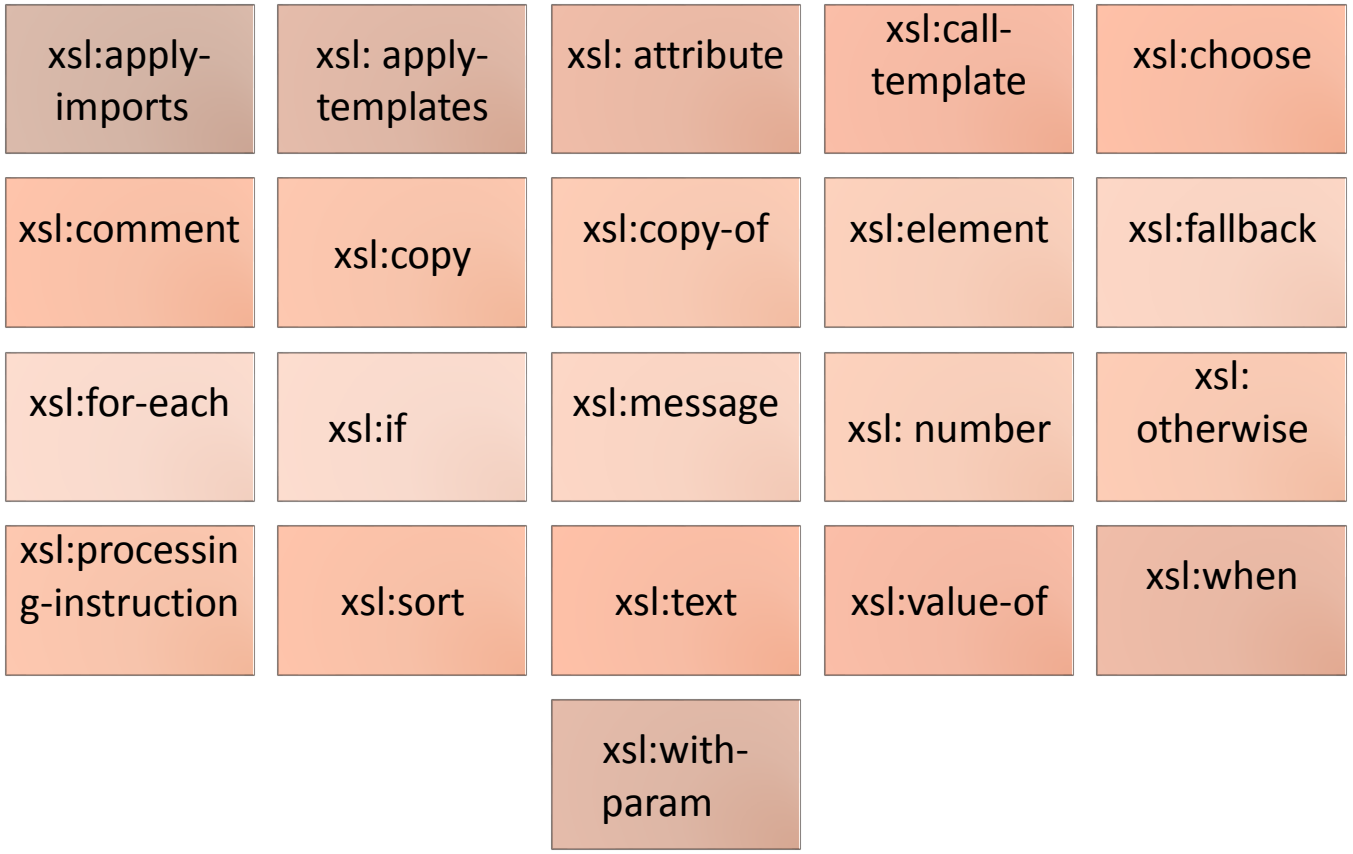
top-level elements

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
…templates and transformation rules go here …
</xsl:stylesheet>
```

Structure of the XSLT document

xsl:template	• defines the template
xsl:variable	• defines variables
xsl:output	• determines the type of result document
xsl:attribute-set	• it defines a set of attributes
xsl:decimal-format	• defines how the numbers are displayed
xsl:import	• import style sheets
xsl:include	• include style sheets
xsl:key	• key that can be used in the style sheet with the key() function
xsl:namespace-alias	• indicates the namespace in the resulting document
xsl:param	• allows you to create parameters
xsl:preserve-space	• retains white characters in the specified element
xsl:strip-space	• removes whitespace from the specified element

Structure of the XSLT document



The format of the output document

xsl:output

method

- defines the output format: html, xml, text
- default method: xml

encoding, *version*, …

XSLT extraction of information

xsl:value-of

select

- determines the node from which we want to get the information
- the attribute contains an XPath expression or XSLT function

only **first node** from set

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

XSLT - templates

xsl:template		
A template contains rules to apply when a specified node is matched.	The template can be called by matching the node of the XML document tree to the *match* attribute	Nodes are **matched** and not selected, they are processed as they are encountered by the XML processor

XSLT - templates

xsl:template

match	*name*	*mode*	*priority*
• pattern which signifies the element(s) on which template is to be applied	• name of the element on which template is to be applied	• allows element to be processed multiple times to produce a different result each time	• priority number of a template

template

```
<xsl:template match= "XPath-expression" name="name”>
  …
</xsl:template>
```

hello.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type= ”text/xsl” href= ”hello.xsl”?>

<hi>Hello XSLT!</hi>
```

hello.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl=“http://www.w3.org/1999/XSL/Transform”  version="1.0">
   <xsl:output method="html" />
   <xsl:template match="/">
      <xsl:value-of select="hi"/>
    </xsl:template>
</xsl:stylesheet>
```

XSLT - template

A series of rules (called template rules)

- Each rule is a sequence of XSLT commands
- Each command is an XML element with attributes

A rule is executed when it

- matches some condition
- or is called by name

XSLT - templates

the <template> element must have one of the attributes:

match

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

defines the transformation for the nodes described by this attribute

name

<xsl:template name="...">
...
</xsl:template>

gives the template a name

XSLT - templates

template with *match* attribute

- template definition:

```
<xsl:template match="node">...</xsl:template>
```

using template:

- all of the children of the current node

```
<xsl:apply-templates/>
```

- selecting the nodes to which the template is used

```
<xsl:apply-templates select="node" />
```

XSLT - templates

template with *name* attribute

- template definition:

```
<xsl:template name="dosomething">...</xsl:template>
```

using template :

-

```
<xsl:call-template name=" dosomething "/>
```

looks like a traditional programming language

we have full control over the way of execution the code

XSLT - templates

```
...
<xsl:template match="mammals">
  <xsl:apply-templates/>
</xsl:template>
...
<xsl:template match="birds">
  <xsl:apply-templates/>
</xsl:template>
…
```

```
...
<xsl:template match="mammals | birds">
  <xsl:apply-templates/>
</xsl:template>
...
```

XSLT - templates

If no rule matches the node?
built-in template

for document root and elements:	for text nodes and attributes:	for comments and processing instructions
apply templates to children	*copy text value to result*	*do nothing*

XSLT - templates

template with mode

- possibility to process the same set of nodes many times

```
<xsl:template match="root">
    <xsl:apply-templates select= ”elem” mode="tryb1"/>
    <xsl:apply-templates select= ”elem” mode="tryb2"/>
</xsl:template>

<xsl:template match="elem" mode="tryb1">
    <xsl:text>tryb1:</xsl:text>
    <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="elem" mode="tryb2">
    <xsl:text>tryb2:</xsl:text>
    <xsl:value-of select="."/>
</xsl:template>
```

Resolving matching rules conflicts

Which template will be used?

there is no template for the given context
- built-in template will be used

there is one template for the given context
- this template will be used

there are 2 templates for the given context
- a more specific template will be applied

there are 2 templates for the given context, both equally specific
- a template with higher priority will be applied

there are 2 templates: the same context, priority, both equally specific
- a template that appears later will be applied

Example

```
<Root>
        <x>1</x>
        <y>5</y>
        <x>2</x>
        <x>3</x>
        <y>6</y>
        <x>4</x>
</Root >
```

```
<xsl:template match="/">
        <xsl:apply-templates />
</xsl:template>
<xsl:template match="y">
        <xsl:value-of select="." />
</xsl:template>
<xsl:template match="x">
        <xsl:value-of select="." />
</xsl:template>
```

Example

```
<Root>
        <x>1</x>
        <y>5</y>
        <x>2</x>
        <x>3</x>
        <y>6</y>
        <x>4</x>
</Root >
```

```
<xsl:template match="/">
        <xsl:apply-templates select="Root/x" />
</xsl:template>
<xsl:template match="y">
        <xsl:value-of select="." />
</xsl:template>
<xsl:template match="x">
        <xsl:value-of select="." />
</xsl:template>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="p1.xsl" ?>
<pajeczaki>
  <pajaki gatunek="tygrzyk"  chroniony="tak" wyst_polska="tak">
    <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
    <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
    <opis czego_dotyczy="Cechy">
      Samica: jej głowotułów pokryty jest gęstym srebrzystym włosem, a na jas
    </opis>
    <opis czego_dotyczy="Pokarm">
      Pająk ten jest bardzo "wybredny", gdyż poluje tylko na owady z rzędu pr
    </opis>
    <zdjecie zrodlo="Gfx\tygrzyk14_obiad.jpg"/>
    <linki href="http://pl.wikipedia.org/wiki/Tygrzyk paskowany">http://pl.wi
  </pajaki>
  <skorpiony gatunek="polny" chroniony="tak" wyst_polska="nie">
    <nazwa jezyk="polska">Skorpion Polny</nazwa>
    <nazwa jezyk="lacińska">Buthus Occitanus</nazwa>
    <opis czego_dotyczy="Cechy">
      Posiadając małe kleszcze i gruby ogon, ten bardzo jadowity gatunek jest
    </opis>
    <opis czego_dotyczy="Poka
      Osobnik należy do niezw
    </opis>
    <zdjecie zrodlo="Gfx/skor
    <linki href="http://www.a
  </skorpiony>
</pajeczaki>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:msxsl="urn:schemas-microsoft-com:xslt" exclude-result-prefixes="msxsl">
    <xsl:output method="html" />

    <xsl:template match="nazwa">
      <strong>
        <xsl:value-of select="."/>
      </strong>
      <br/>
    </xsl:template>

</xsl:stylesheet>
```

XSLT

The XSLT language allows to		
conditional processing,	looping	parameterisation

XSLT - conditional processing

xsl:if

- **test**

no else

```
<xsl:if   test="gender='woman‘ ">
            <font class="napis">Wife</font>
</xsl:if>
<xsl:if   test="gender='man‘ ">
            <font class="napis">Husband</font>
</xsl:if>
```

XSLT

xsl:choose : conditional processing

- xsl:when : determines the condition
  - test
- xsl:otherwise : optional, final instructions, used if no condition satisfied
- only first branch with satisfied condition processed.

```
<xsl:choose>
        <xsl:when test="position()=first()">
                Do something for first element
        </xsl:when>
        <xsl:when test="position()=last()">
                Do something for last element
        </xsl:when>
        <xsl:otherwise>
                Do something for other elements
        </xsl:otherwise>
</xsl:choose>
```

XSLT - loop

xsl:for-each

- **select**

the XSLT processor processes all nodes corresponding to the pattern given in the *select* attribute

possibility of sorting - xsl:sort

```
<xsl:for-each select="XPath expression">
        some instructions
</xsl:for-each>
```

Sorting

xsl:sort
attributes

- **select**
  - sorting by element or attribute
- **order**
  - *ascending*, *descending*
- **case-order**
  - specifies letter size priority, *upper-first* , *lower-first*
- **lang**
- **data-type**
  - sorting letters or numbers (*text*, *number*)

possibility of multiple sorting criteria

xsl:sort

- xsl:for-each
- xsl:apply-templates

xsl:sort the first instruction in for-each

```
<xsl:for-each select="osoba">
  <xsl:sort select="@plec"/>
  <xsl:sort select="nazwisko"/>
  <xsl:sort select="imie"/>
    <center>
    <xsl:choose>
      <xsl:when test="@plec='kobieta'">
            <table class="kobieta" align="center"> <xsl:call-template name="tabelka"/>
            </table>
      </xsl:when>
      <xsl:when test="@plec='mezczyzna'">
            <table class="mezczyzna" align="center"> <xsl:call-template name="tabelka"/>
            </table>
      </xsl:when>
    </xsl:choose>
    </center>
</xsl:for-each>
```

Numbering

xsl:number

it can be located anywhere in the templates also in for-each element

defining the form of the number - attribute *format*

- 1 1,2,3...
- 01 01,02,03,...
- a a,b,c,...z,aa...
- B A,B,C...
- i i,ii,iii,iv,...
- ....

Numbering

level attribute

- = ”any”
  - continuous numbering of elements regardless of their parent element
- = ”multiple”
  - multi-level numbering (eg 1.2; 3.2.4, …)

the possibility of grouping numbering

- grouping-separator , grouping-size
  - grouping-separator=”.” , grouping-size=”3”
  - 12345678 -> 12.345.678

```
- <linki>
    <link adres="http://pl.wikipedia.org/wiki/Tygrzyk_paskowany">wikipedia.org</link>
    <link adres="http://www.salamandra.org.pl/magazyn/b08a07.html">salamandra.org.pl</link>
    <link adres="http://www.eko.org.pl/otop-leszno/tygrzyk.php">eko.org.pl</link>
    <link adres="http://www.bocian.org.pl/biuletyn/2001/b2a23.php">bocian.org.pl</link>
    <link adres="http://grzesiak.kei.pl/jurek/lista161.html">grzesiak.kei.pl</link>
  </linki>

- <xsl:for-each select="media/linki/link">
    <xsl:sort select="." order="descending" />
    <xsl:number format="1." />
    <xsl:text />
  - <a target="_blank" href="{@adres}">
      <xsl:value-of select="." />
    </a>
    <br />
  </xsl:for-each>
```

---

1. wikipedia.org
2. salamandra.org.pl
5. grzesiak.kei.pl
3. eko.org.pl
4. bocian.org.pl

```
- <xsl:for-each select="media/linki/link">
    <xsl:sort select="." order="descending" />
    <xsl:number value="position()" format="1." />
    <xsl:text />
  - <a target="_blank" href="{@adres}">
      <xsl:value-of select="." />
    </a>
    <br />
  </xsl:for-each>
```




---

1. wikipedia.org
2. salamandra.org.pl
3. grzesiak.kei.pl
4. eko.org.pl
5. bocian.org.pl

Variables

xsl:variable

definition

- <xsl:variable name="VarName">value</xsl:variable>
- <xsl:variable name="VarName" select ="'’value’'"/>

reference to variable

- <xsl:value-of select=”$VarName”/>

Variables

local

```
<xsl:template name="...">
        <xsl:variable name="...">...</xsl:variable>
</xsl:template>
```

global

```
<xsl:stylesheet ...>
        <xsl:variable name="...">...</xsl:variable>
        ...
</xsl: stylesheet >
```

Variables

simple

- they contain single values
- usually used to insert the same values in many places in a document

complex

- they contain sets of nodes and tree fragments

Variables??

Constant (*name given to a certain value*)

their values can not be modified (*read only*)

advantages of variables

- they make reading the code easier
  - complex expression saved as a variable
  - the ability to break complex expressions into parts
- reuse
  - performance enhancement especially for complex expressions that result in a fragment of the tree
- saving the value of nodes that are currently unavailable

```
<xsl:template match="zwierzak">

        <!-- tytuł: -->
        <h2><xsl:value-of select="nazwa/polska"/>, (<i><xsl:value-of select="nazwa/lacinska"/></i>)</h2>
        <img align="left" hspace="10" src="img/archeopteryks/tb/archeopteryks_05.jpg"/>
        <xsl:apply-templates select="wystepowanie"/> <!-- szablon wystepowania -->
        <xsl:apply-templates select="opis"/> <!-- szablon opisu -->

        <xsl:variable name="nazwa_zwierza">     <!-- zmienna nazwy -->
                <xsl:value-of select="nazwa/polska"/>
        </xsl:variable>


        <h4>Zdjęcia:</h4>
                <xsl:for-each select="zdjecia">
                        <img src="{.}" alt="{$nazwa_zwierza}">
                                <xsl:comment><xsl:value-of select="."/></xsl:comment>
                        </img>
                </xsl:for-each>
        <br/>

        <xsl:if test="position()!=last()">      <!-- linie rozdzielaj±ce zwierzaki -->
                <br/><br/><hr/><br/>
```

Opis:

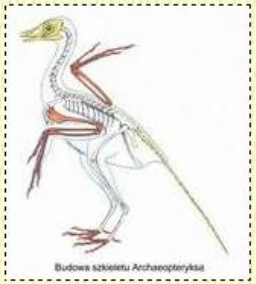
- Wygląd: Duży, masywny ptak o uwstecznionych skrzydłach z charakte
  biały.
- Dodatkowe informacje: Pierwszy raz opisany został przez żeglar
  posiadający masywne, krótkie nogi i uwstecznione skrzydła i ogon.
- Pokarm: Brak dostatecznie wiarygodnych danych. Według przekazu
  że żywił się kamieniami i żelazem (prawdopodobnie dodo połykały ka
- Wymiary:
  - Długość ciała: 75 cm
  - Masa: 20 kg

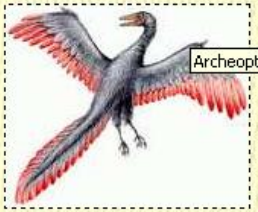
Zdjęcia:

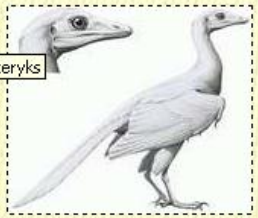
Opis:

- Wygląd: Wygląd stanowi zagadkę, jak to u zwierzą kopalnych bywa. Wiadomo, że posiadał p
  ptaka.
- Dodatkowe informacje: Gatunek znany jest z sześciu skamielin znalezionych w litografic
  okolicach miejscowości Solnhofen w Bawarii.
- Pokarm: Owady
- Wymiary:
  - Długość ciała: 45 cm
  - Masa: 0.3 kg

Zdjęcia:







```
<xsl:variable name="bgcolor">
  <body>#cccccc</body>
  <table>#ffffff</table>
  <row>#cccccc</row>
  <altrow>#ffffff</altrow>
</xsl:variable>

<xsl:template match="/">
  <html>
  <body bgcolor="{$bgcolor/body}">
    <xsl:apply-templates />
  </body>
  </html>
</xsl:template>

<xsl:template match="cars">
  <table bgcolor="{$bgcolor/table}" width="75%">
    <xsl:for-each select="car">
      <tr>
        <xsl:attribute name="bgcolor">
          <xsl:choose>
            <xsl:when test="position() mod 2 = 0">
              <xsl:value-of select="$bgcolor/altrow" />
            </xsl:when>
            <xsl:when test="position() mod 2 = 1">
              <xsl:value-of select="$bgcolor/row" />
            </xsl:when>
          </xsl:choose>
```

Focus	Ford	2000
Golf	Volkswagen	1999
Camry	Toyota	1999
Civic	Honda	2000
Prizm	Chevrolet	2000

Formatting numbers

Converting numerical values into strings

format-number(number, format_pattern, dec_format)

value to format

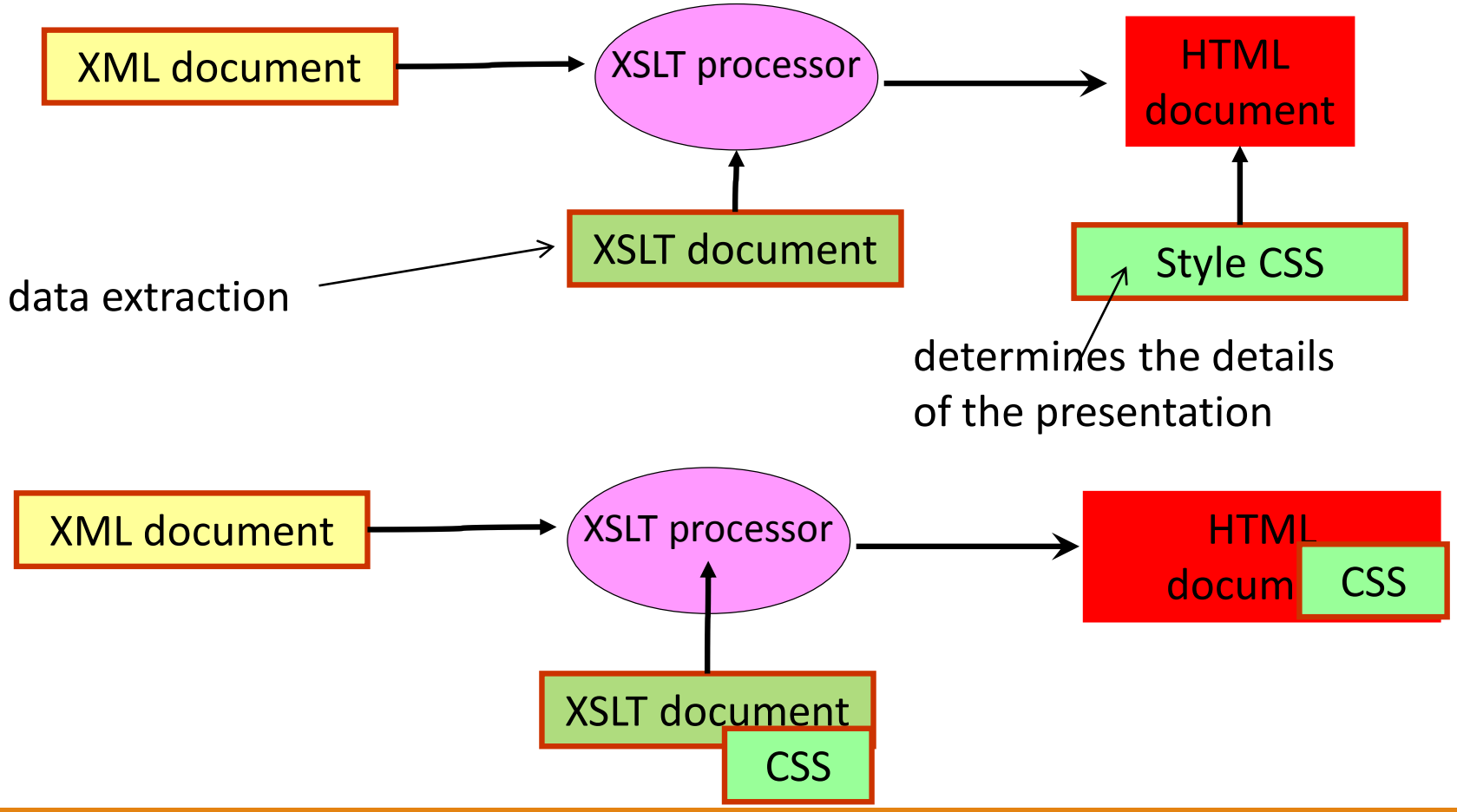
name of the user-defined format

{prefix}number{.fraction}{suffix}

0 Digit
Digit, zero shows as absent
. The position of the decimal point (###.##)
, The group separator for thousands (###,###.##)
% Displays the number as a percentage (##%)

```
<xsl:decimal-format
    name="formatname"
    decimal-separator=""
    grouping-separator=""
    infinity=""
    minus-sign=""
    NaN=""
    percent=""
    per-mille=""
    zero-digit=""
    digit=""
    pattern-separator=""/>
```

XSLT – using CSS



Changing the structure of the output document

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xml-stylesheet type="text/xsl" href="Pajeczaki2.xsl"?>
<pajeczaki xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <pajaki gatunek="tygrzyk"  chroniony="tak" wyst_polska="tak">
      <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
      <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
      <gromada>Pajeczaki</gromada>
      <wystepowanie>
        <swiat>...</swiat>
        <tereny>...</tereny>
       <srodowisko typ="lądowe" />
      </wystepowanie>
      <opis czego="Cechy">Samica: jej głowotułów pokryty jest gęstym s
      </opis>
      <opis czego="Pokarm">Pająk ten jest bardzo "wybredny", gdyż polu
      </opis>
      <opis czego="Wymiary">Samica osiąga długość ok. 25 mm. Samiec je
      </opis>
      <zdjecie zrodlo="Gfx\tygrzyk10_mlody2.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk12_ml_samica.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk12_ml_samiec.jpg"></zdjecie>
      <zdjecie zrodlo="Gfx\tygrzyk14_obiad.jpg"></zdjecie>
      <linki adres="http://pl.wikipedia.org/wiki/Tygrzyk_paskowany">wi
      <linki adres="http://www.salamandra.org.pl/magazyn/b08a07.html">
      <linki adres="http://www.eko.org.pl/otop-leszno/tygrzyk.php">eko
      <linki adres="http://www.bocian.org.pl/biuletyn/2001/b2a23.php">
      <linki adres="http://grzesiak.kei.pl/jurek/lista161.html">grzesi
      <data>2012-12-01</data>
   </pajaki>
```

```
<?xml version="1.0" encoding="UTF-16"?>
- <zwierzeta>
  - <tygrzyk>
      <opis>Samica: jej głowotułów pokryty jest gęst
        upodabnia ją nieco do osy. Samiec: jest bar
        bokach. </opis>
      <zdjecie>Gfx\tygrzyk10_mlody2.jpg</zdjecie>
    </tygrzyk>
  - <krzyzak>
      <opis>Na odwłoku jasne plamy układają się w c
        odcień brązu. Cechą charakterystyczną jest
        pozbawiony nóg, przeważnie jest pękaty. Sa
      <zdjecie>Gfx/krzyzak.jpg</zdjecie>
    </krzyzak>
  - <polny>
      <opis>Posiadając małe kleszcze i gruby ogon, t
        mniej jadowita odmiana tego gatunku, char
        należy do najbardziej jadowitych skorpionó
        od 19 do 30, natomiast u samca 25 do 36. P
      <zdjecie>Gfx/skorpion1.jpg</zdjecie>
    </polny>
</zwierzeta>
```

```
<?xml version="1.0" encoding="iso-8859-2"?>
<?xml-stylesheet type="text/xsl" href="p2.xslt"?>
<zwierzeta xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:nc


<gatunek numer="1">
        <data>2005-12-01</data>
        <nazwapl
        <nazwala
        <gromada
        <wystepo


        </wystep
        <opis>


        </opis>
        <czychronione>T</czychronione>
        <czywpolsce>T</czywpolsce>

        <zdjecia src="photo/1.jpg"/>
        <linki>
                <odnosnik src="http://www.kruki.pl"/>
                <odnosnik src="http://www.kruki.com"/>
        </linki>
</gatunek>

<gatunek numer="2">
        <data>2005-12-01</data>
```

Gatunek:	Kruk (*Corvus corax L.*)	Kanarek (*Serinus canaria canaria*)	Zieba (*Fringilla coelebs L.*)
			
Data modyfikacji:	2005-12-01	2005-12-01	2005-12-01
Gromada:	ptaki	ptaki	ptaki
Region wystepowania:	Europa Srodkowa	Wyspy Kanaryjskie	Europa i Azja

```
<xsl:element name="img">
    <xsl:attribute name="src"><xsl:value-of select="@src"/></xsl:attribute>
    <xsl:attribute name="width">150</xsl:attribute>
</xsl:element>
```

```
<xsl:element name=”a”>
        <xsl:attribute name=”href”>
                <xsl:value-of select=”address”/>
        </xsl:attribute>
        <xsl:value-of select=”.”/>
</xsl:element>
```

zdjecia	    
linki	1. http://motyle.biol.uni.torun.pl/atl/t4.htm 2. http://pl.wikipedia.org/wiki/Niepylak_apollo 3. http://robale.pl/index/2/186 4. http://www.zzw-niedzica.com.pl/niepylak.htm

```
<document>
        ...to a physical condition of a do
        may also be blurred with stains,
        corners and other noise-like effe
    <paragraph/>
        In order to tackle these problem
        degrees of fatigue.
    <paragraph/>
        A set of selected documents may be satisfactory for some of
        these aspects, and at the same time ... defined quality metrics to
        measure the indicated document aspects.
    <paragraph/>
        A methodology for measuring quality of docu
        phases is...was used with relative wide band
    <paragraph/>
        Other issues...can be worked out with the QE
    <paragraph/>
        Quality improvement that can be really obtai
        adding to the DDLC...
</document>
```

```
<xsl:template match="paragraph">
    <p>
    -------
    </p>
</xsl:template>
```

```
<xsl:template match="paragraph">
    <xsl:element name="p">
    -------
    </xsl:element>
</xsl:template>
```

Creating new elements

Element or attribute name established at runtime

```
<xsl:template match="……..">
  <xsl:element name="name_of_the_element">
  -------
  </xsl:element>
</xsl:template>
```

```
<xsl:template match="……..">
  <xsl:element name="{node_of_the_document_tree}">
  -------
  </xsl:element>
</xsl:template>
```

```
<xsl:element name=”img”>
   <xsl:attribute name=”src”>
      <xsl:value-of select= ”@zrodlo”/>
   </xsl:attribute>
   <xsl:attribute name=”alt”>
      photo of the city
   </xsl:attribute>
</xsl:element>
```

```
<img src=” {@zrodło}” alt=”photo of the city”/>
```

Text

we write the text

<xsl:text>

```
<xsl:value-of select="nazwisko">
<xsl:text>   </xsl:text>
<xsl:value-of select="imie">
<xsl:text>   </xsl:text>
```

Push-me Pull-you Stylesheets

INPUT-DRIVEN (CALLED PUSH)

walk the input tree

match elements in input tree

do something when you find a match

STYLESHEET DRIVEN (CALLED PULL)

more like a typical computer program

walk the stylesheet (which specifies the order of the output document)

when it asks for data, go get it from the input tree

Push-me Pull-you Stylesheets

Input-driven (called Push) Stylesheet driven (called Pull)

Why Programmers Will Like Pull?

They are used to controlling the order of execution

They don't trust the stylesheet to "do it right"

They need recursion, so they force it (instead of using it)

They fight the template design

Why Pull Can Be a Problem?

Stylesheets tell the processor what to do when it finds something. The processor controls the finding of things in the input tree (when to do it).

Pull stylesheets:control both what and when, and fight the design of the language

Pull: The Big Beginner Mistake

Pull should be used sparingly for special situations only

- best used on very regular, predictable structures (data)

Beginners use it for everything

How to notice this problem in stylesheets

- using <xsl:for-each> to force recursion (instead of using templates, which recurse as designed)
- using
  - <xsl:value-of> which processes the text inside an element
  - instead of <xsl:apply-templates>, which processes both text and the embedded tags inside an element

XSLT summary

declarative, data-driven language of transforming XML trees

the basic processing paradigm is pattern matching

Possibility of conditional processing, looping, parameterization, sorting, numbering, ...

Various types of output documents