Technologie založené na XML

XPath, XSLT

Martin Klíma, Ladislav Čmolík

Computer Graphics Group

Zopakování z minula

- XML
  - elementy
  - atributy
  - entity
  - namespace

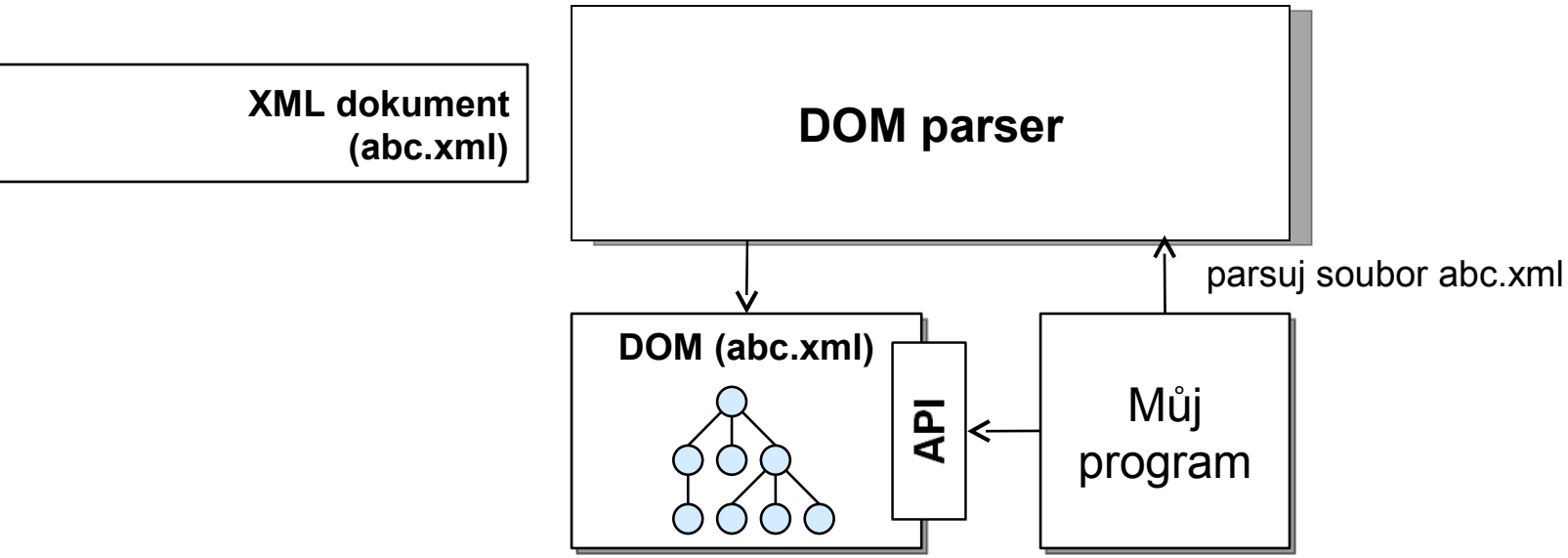
- DTD

Nástroje pro zpracování XML (parsery)

- 2 přístupy
  - DOM parser (DOM = Document Object Model)
  - SAX parser (SAX = Simple API for XML)

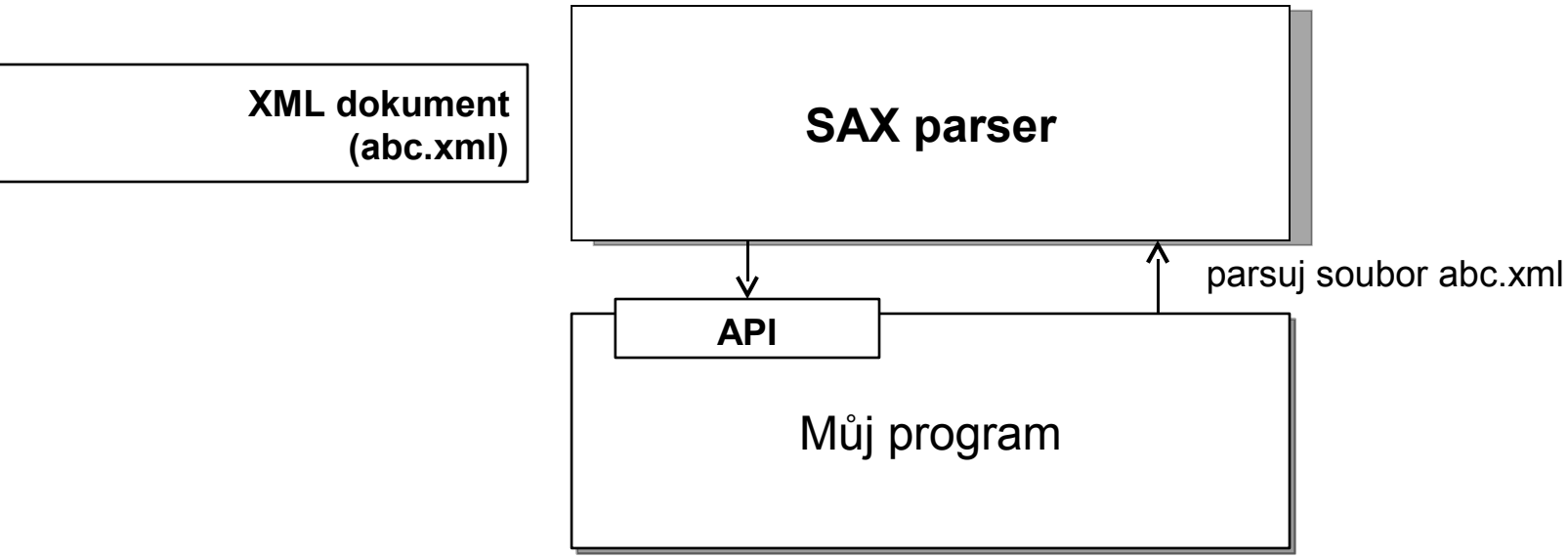
- DOM parser vezme XML dokument a vyrobí jeho obraz v paměti (DOM)
- SAX parser postupně prochází XML soubor a vyhazuje události. Je na programátorovi, aby tyto události zpracoval.

DOM parser



- V paměti počítače je vytvořen DOM celého XML dokumentu
- DOM lze modifikovat (přidávat a odebírat elementy, atributy, atd.) a opět uložit jako XML dokument
- V DOMu lze vyhledávat pomocí jazyka XPath
- Snadná kontrola XML podle DTD

SAX parser



- SAX parser do paměti nic neukládá pouze zasílá události
  - Je tedy možné zpracovávat libovolně veliké XML dokumenty
- V době vyslání události se neví zda je XML dokument validní
  - To je známo až po zpracování celého XML dokumentu
- DOM parser používá SAX parser pro vytvoření DOMu

XPath

- Je to jazyk pro hledání (adresaci) informace v XML dokumentu
- Vznikl jako součást W3C doporučení XSLT
- Na XPath staví další jazyky jako XPointer a XQuery

- XPointer je jazyk pro provázání dokumentů, umožňuje ukázat na konkrétní část nějakého XML dokumetu
- XQuery je jazyk pro dotazování nad XML daty. Je to obdoba SQL nad relačními databázemi

XPath

- Je to syntaxe pro pojmenovávání (adresování) částí XML dokumentů
- Vytváří výrazy pro navigaci v XML dokumentu
- Obsahuje sadu funkcí (více než 100)
- Je důležitou součástí XSLT
- Je to doporučení W3C

XPath – Datový model

XPath modeluje XML dokument jako strom

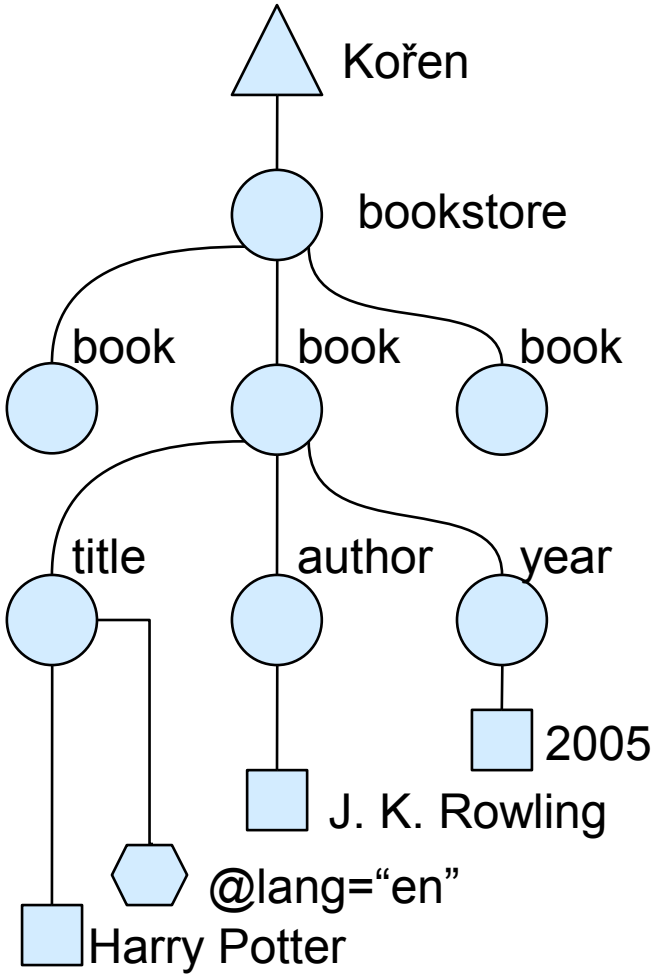
- Není to DOM

Datový model obsahuje uzly:

Typ uzlu	Označení uzlu v XPath
Kořen (root) POZOR: Kořen není totéž co kořenový element!!!	/
Element	jméno_elementu, * = libovolný element
Element ve jmenném prostoru	prefix_jmeného_prostoru:jméno_elementu
Atribut	@jméno_atributu, @* = libovolný atribut
Text	text()
Instrukce	processing-instruction()
Komentář	comment()
	node() = libovolný typ uzlu

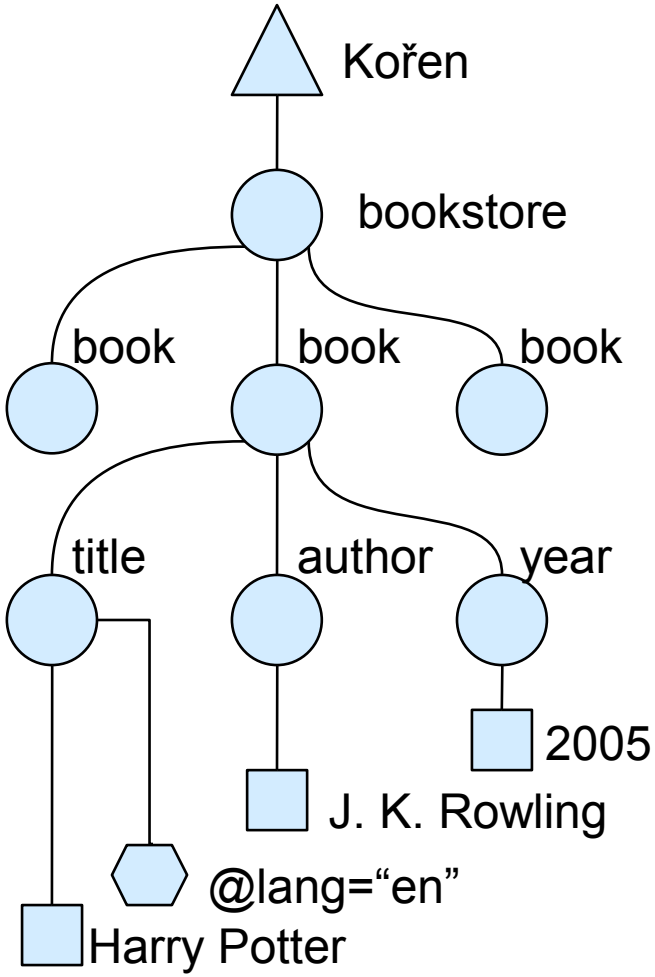
XPath – Datový model - Příklad

```
<?xml version="1.0"
    encoding="UTF-8"?>
<bookstore>
    <book></book>
    <book>
      <title lang="en">
          Harry Potter
      </title>
      <author>J. K. Rowling</author>
      <year>2005</year>
    </book>
    <book></book>
</bookstore>
```



XPath – Datový model

- Co v datovém modelu není vidět
  - pořadí atributů, jak byly napsány
  - entity a kódy znaků
  - CDATA

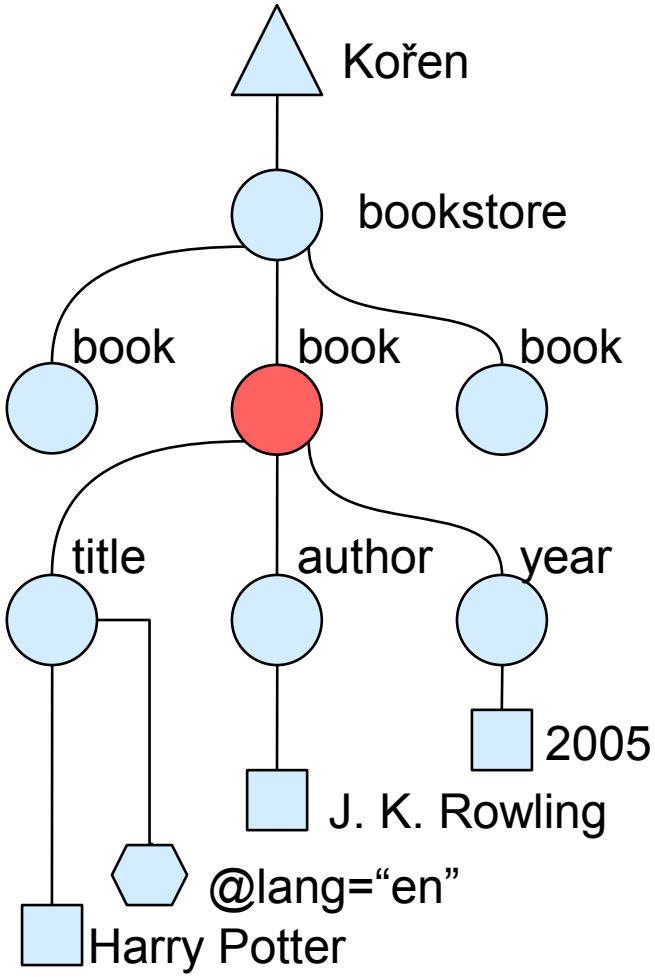


Co tedy XPath přesně dělá

- Vyhodnocuje zadaný *XPath výraz* na datovém modelu XML dokumentu
  - Výsledek je výběr, tj. podmnožina uzlů datového modelu
  - …nebo syntaktická chyba ☺
- Uzly lze adresovat
  - Relativně vůči aktuálnímu uzlu
  - Absolutně od kořene

XPath - Relativní adresování uzlů

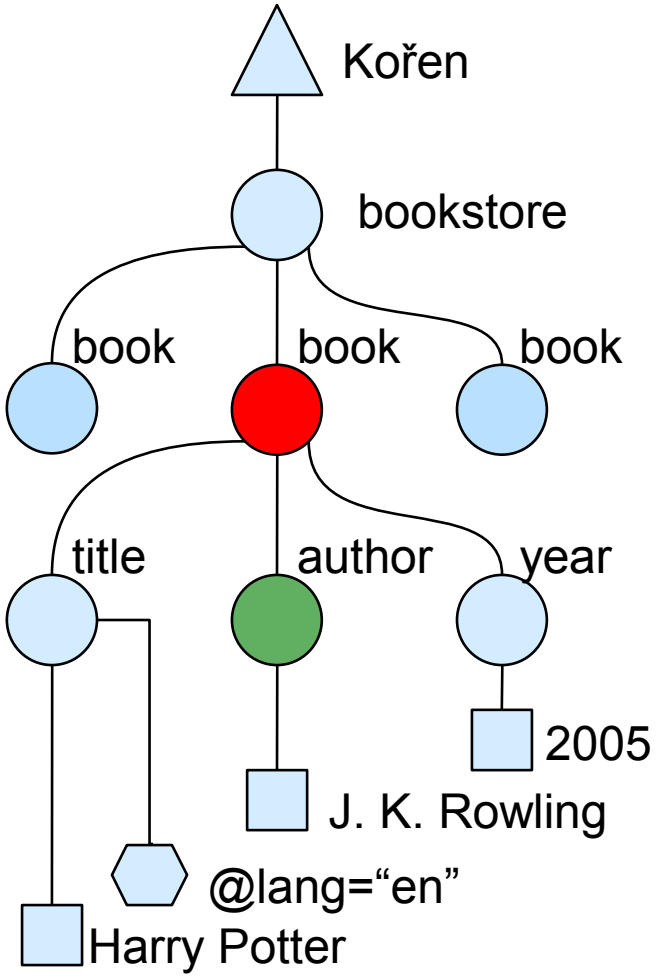
Výraz	Význam
jméno_uzlu	vybere **všechny děti aktuálního uzlu** s daným jménem
//jméno_uzlu	vybere **všechny potomky aktuálního uzlu** s daným jménem
.	vybere **aktuální uzel**
..	vybere **rodiče aktuálního uzlu**
jméno_uzlu_1/ jméno_uzlu	vybere **všechny děti** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1
jméno_uzlu_1// jméno_uzlu_2	vybere **všechny potomky** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1



Computer Graphics Group

XPath - Relativní adresování uzlů

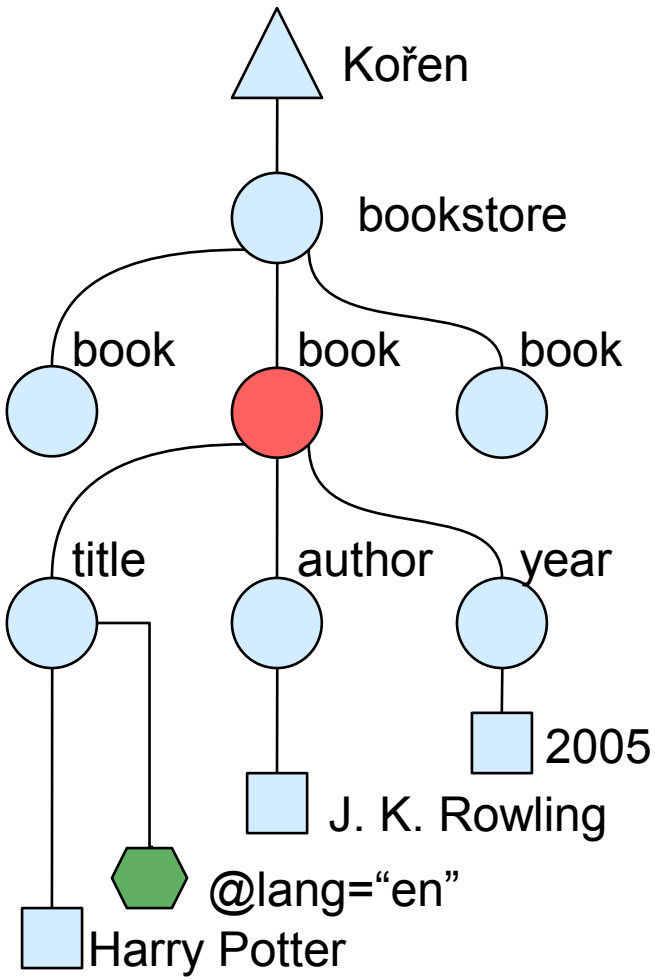
Výraz	Význam
jméno_uzlu	vybere **všechny děti aktuálního uzlu** s daným jménem
//jméno_uzlu	vybere **všechny potomky aktuálního uzlu** s daným jménem
.	vybere **aktuální uzel**
..	vybere **rodiče aktuálního uzlu**
jméno_uzlu_1/ jméno_uzlu_2	vybere **všechny děti** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1
jméno_uzlu_1// jméno_uzlu_2	vybere **všechny potomky** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1



Příklad 1: author

XPath - Relativní adresování uzlů

Výraz	Význam
jméno_uzlu	vybere **všechny děti aktuálního uzlu** s daným jménem
//jméno_uzlu	vybere **všechny potomky aktuálního uzlu** s daným jménem
.	vybere **aktuální uzel**
..	vybere **rodiče aktuálního uzlu**
jméno_uzlu_1/ jméno_uzlu_2	vybere **všechny děti** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1
jméno_uzlu_1// jméno_uzlu_2	vybere **všechny potomky** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1

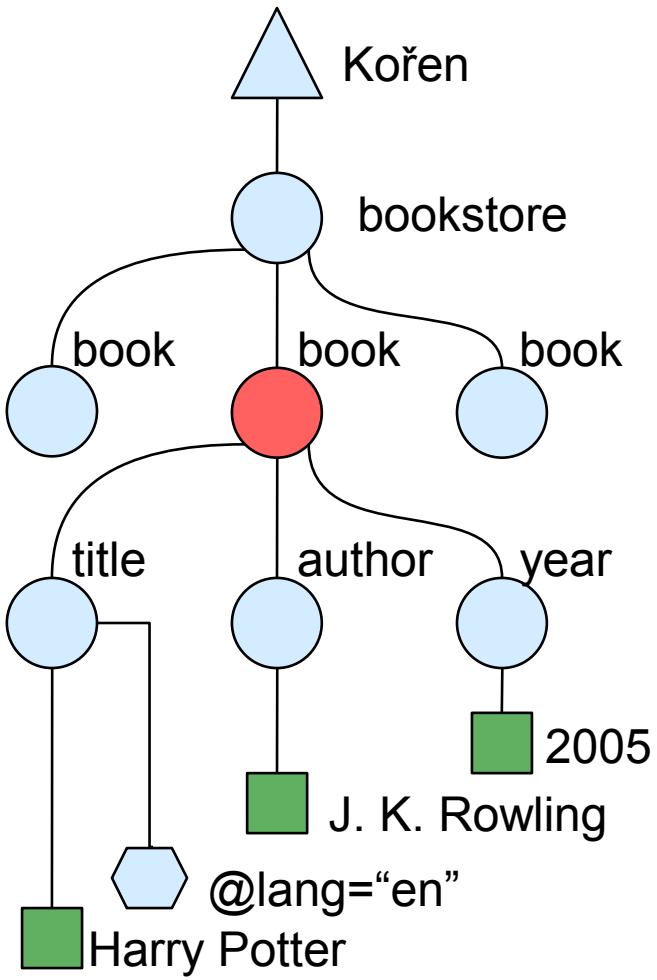


Příklad 2: //@lang

Computer Graphics Group

XPath - Relativní adresování uzlů

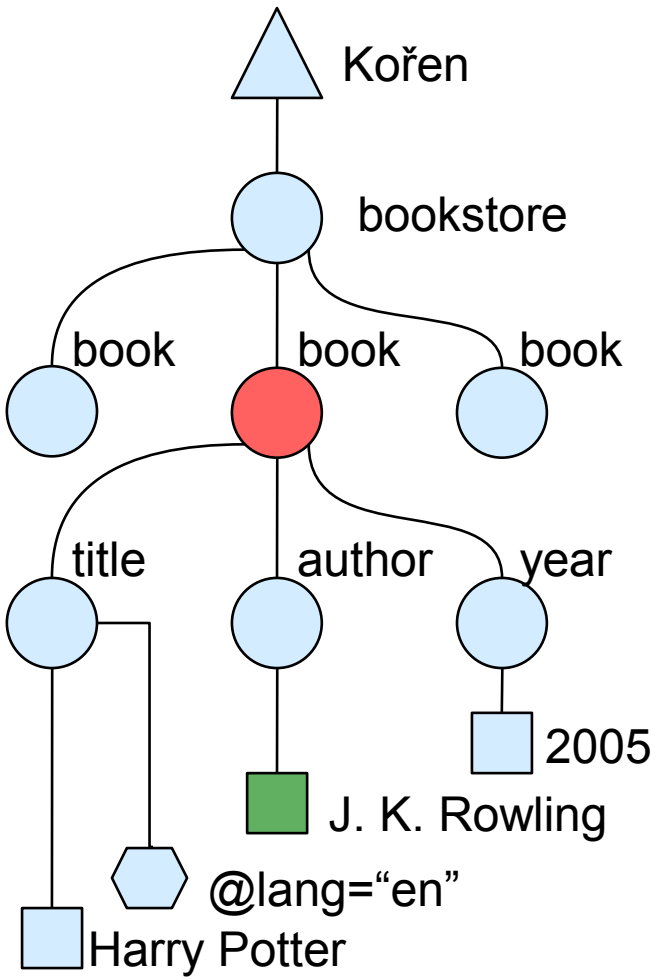
Výraz	Význam
jméno_uzlu	vybere **všechny děti aktuálního uzlu** s daným jménem
//jméno_uzlu	vybere **všechny potomky aktuálního uzlu** s daným jménem
.	vybere **aktuální uzel**
..	vybere **rodiče aktuálního uzlu**
jméno_uzlu_1/ jméno_uzlu_2	vybere **všechny děti** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1
jméno_uzlu_1// jméno_uzlu_2	vybere **všechny potomky** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1



Příklad 3: //text()

XPath - Relativní adresování uzlů

Výraz	Význam
jméno_uzlu	vybere **všechny děti aktuálního uzlu** s daným jménem
//jméno_uzlu	vybere **všechny potomky aktuálního uzlu** s daným jménem
.	vybere **aktuální uzel**
..	vybere **rodiče aktuálního uzlu**
jméno_uzlu_1/ jméno_uzlu_2	vybere **všechny děti** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1
jméno_uzlu_1// jméno_uzlu_2	vybere **všechny potomky** se jménem jméno_uzlu_2 **všech dětí aktuálního uzlu** se jménem jméno_uzlu_1



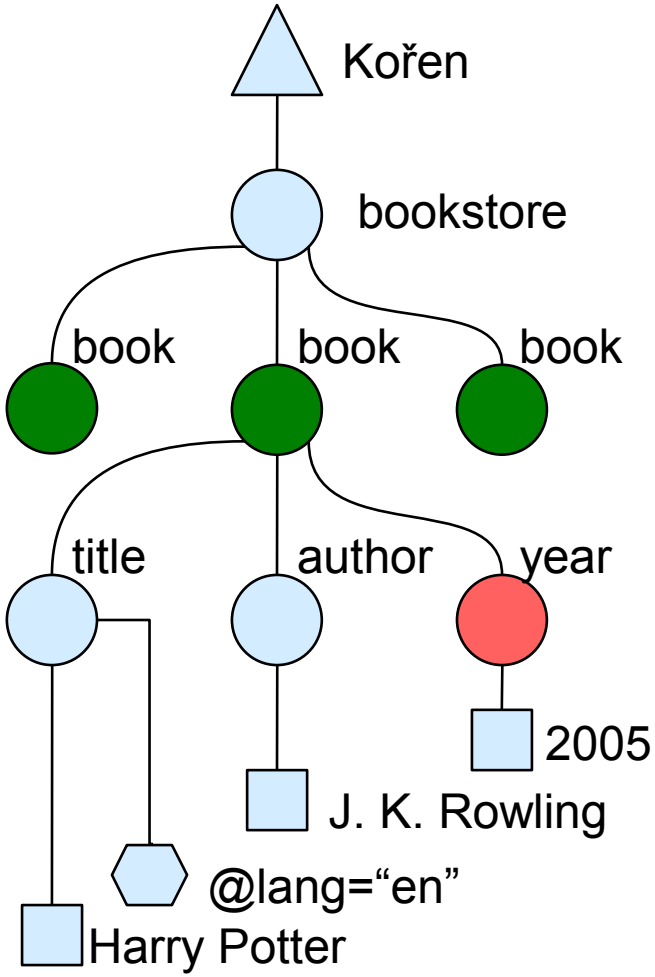
Příklad 4: author/text()

XPath - Absolutní adresování uzlů

- Stejné jako relativní adresování
  - Jako kdyby byl kořen aktuální uzel
  - Všechny výrazy začínají /

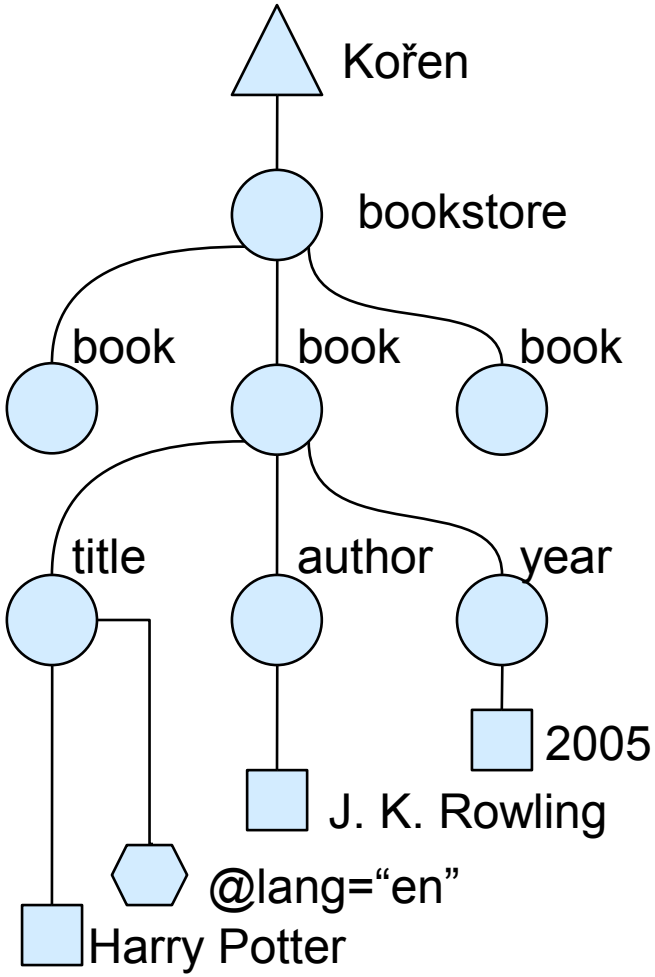
Příklad: /bookstore/book

```
<book></book>
<book>
  <title lang="en">
       Harry Potter
  </title>
  <author>J. K. Rowling</author>
  <year>2005</year>
</book>
<book></book>
```



XPath – Datový model - Příklad

```
<?xml version="1.0"
    encoding="UTF-8"?>
<bookstore>
    <book></book>
    <book>
      <title lang="en">
          Harry Potter
      </title>
      <author>J. K. Rowling</author>
      <year>2005</year>
    </book>
    <book></book>
</bookstore>
```



XPath - Predikáty

- Používají se k výběru uzlu s význačnou vlastností

Výraz	Výsledek
/bookstore/book[1]	Vybere se první element book, který je dítě kořenového elementu bookstore.
/bookstore/book[last()]	Vybere poslední element book, ...
/bookstore/book[last()-1]	Vybere předposledního element book, ...
/bookstore/book[position()<3]	Vybere první dva elementy book, které jsou děti kořenového elementu bookstore.
//title[@lang]	Všechny elementy title s atributem lang, který je potomek aktuálního uzlu
//title[@lang='eng']	Stejné a navíc s hodnotu eng.
/bookstore/book[price>35.00]	Všechny elementy book, které jsou děti kořenového elementu bookstore a obsahují element price s textovou hodnotu větší než 35.00
/bookstore/book[price>35.	Všechny elementy title, které jsou děti elementů viz předchozí řádek.

XPath - Výběr neznámých uzlů

Zástupný znak	Význam
*	Jakýkoli uzel typu element
@*	Jakýkoli uzel typu atribut
node()	Jakýkoli uzel

Výraz	Význam
/bookstore/*	Všechny děti kořenového elementu bookstore
//*	Všechny potomky aktuálního uzlu (Kořene)
//title[@*]	Všechny elementy title, které mají nějaký atribut a jsou potomky aktuálního uzlu (Kořene)

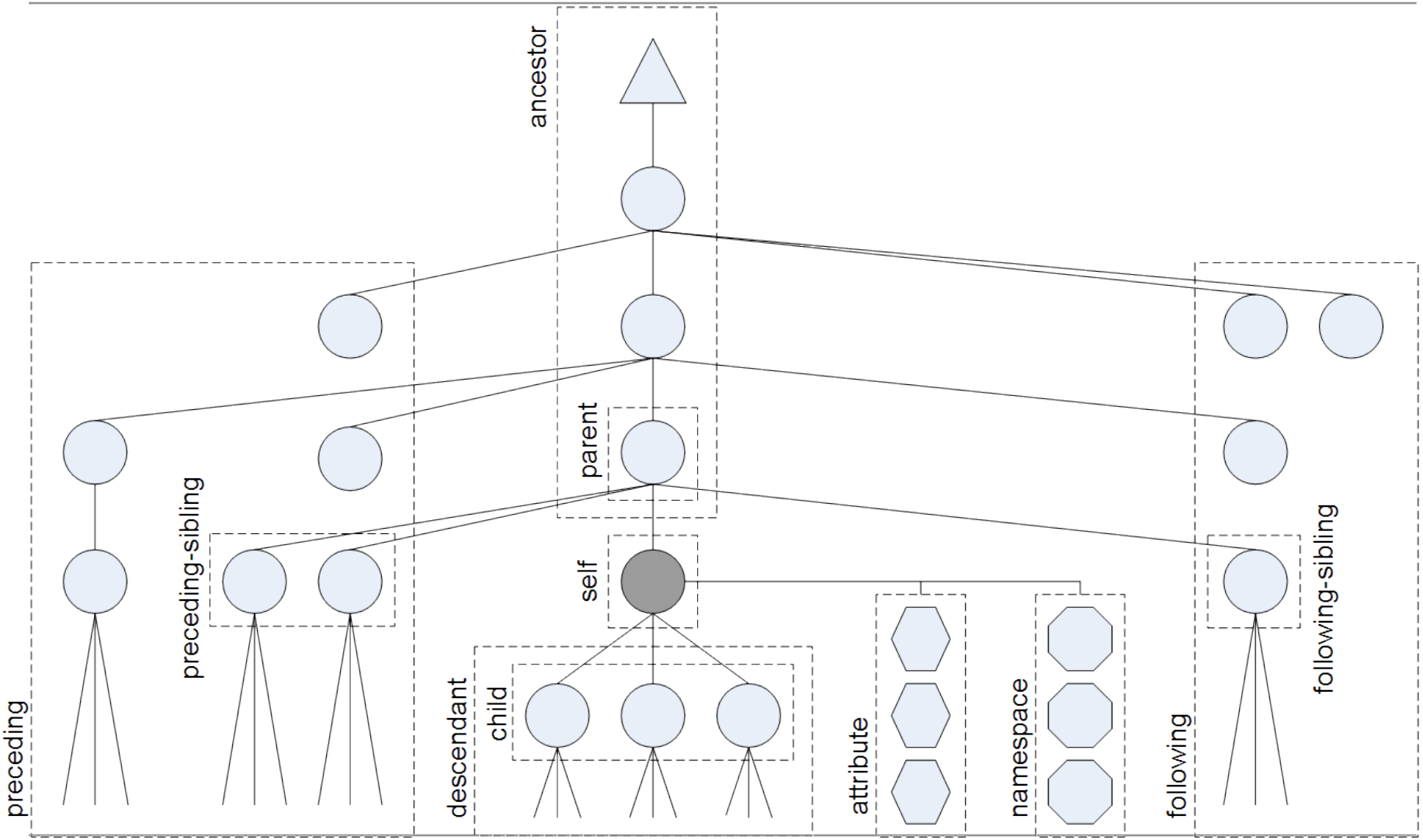
XPath - Výběr více uzlů

Výraz	Význam
//book/title | //book/price	Všechny elemety title pod elementem book a všechny el. price pod el. book. Book je potomkem aktuálního uzlu (Kořene)
//title | //price	Všchny elementy title a price, které jsou potomky aktuálního uzlu (Kořene)

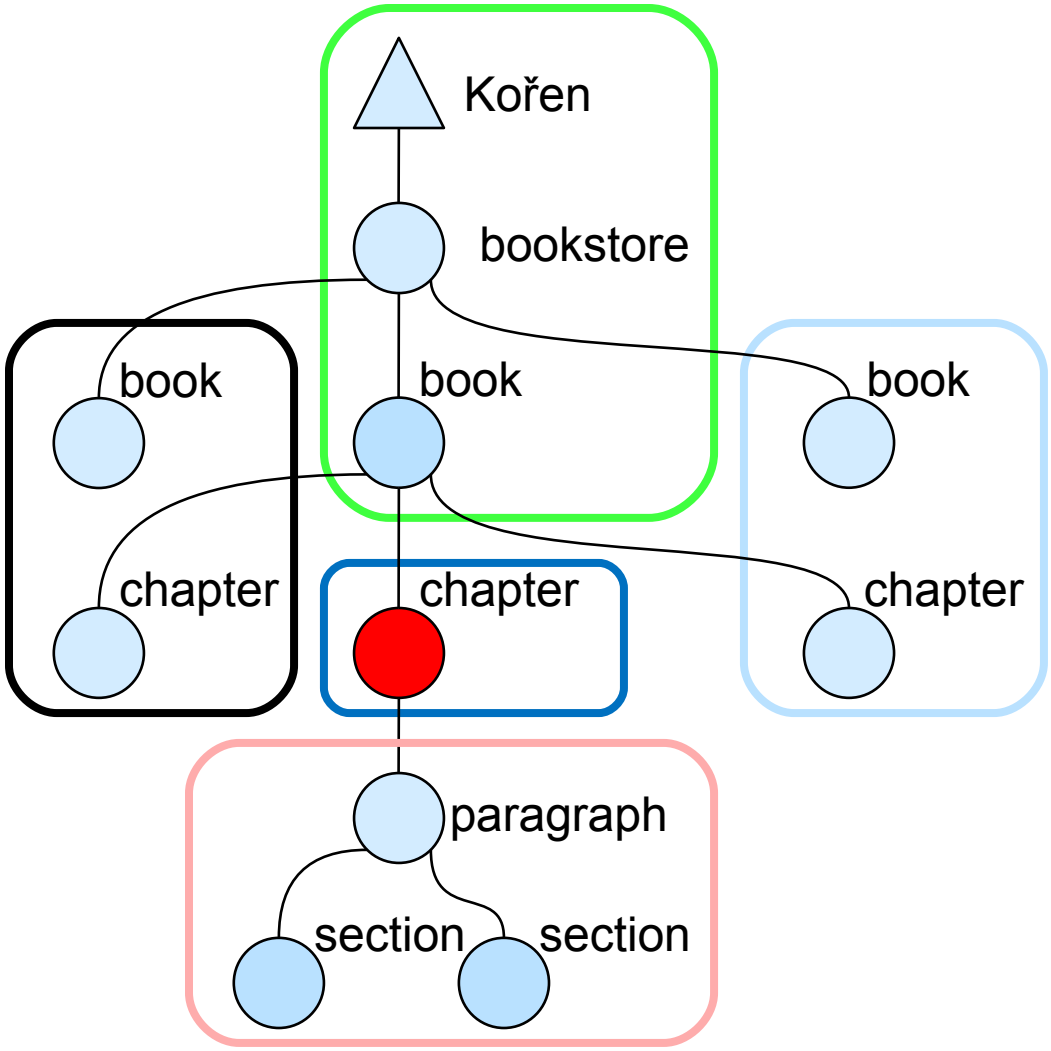
XPath - Axes - Příklady

Výraz	Význam
child::book	Všechny book uzly, které jsou dětmi aktuálního uzlu
attribute::lang	Attribut lang aktuálního uzlu
child::*	Všechny děti aktuálního uzlu
attribute::*	Všechny atributy aktuálního uzlu
child::text()	Všechny děti typu text od aktuálního uzlu
child::node()	Všechny děti aktuálnío uzlu
descendant::book	Všichni potomci book aktuálního uzlu
ancestor::book	Všichni book předchůdci aktuálního uzlu
ancestor-or-self::book	Předchůdci book včetně aktuálního uzlu
child::*/child::price	Všechny uzly price, které jsou vnoučaty aktuálního uzlu

XPath – Axes (Osy)

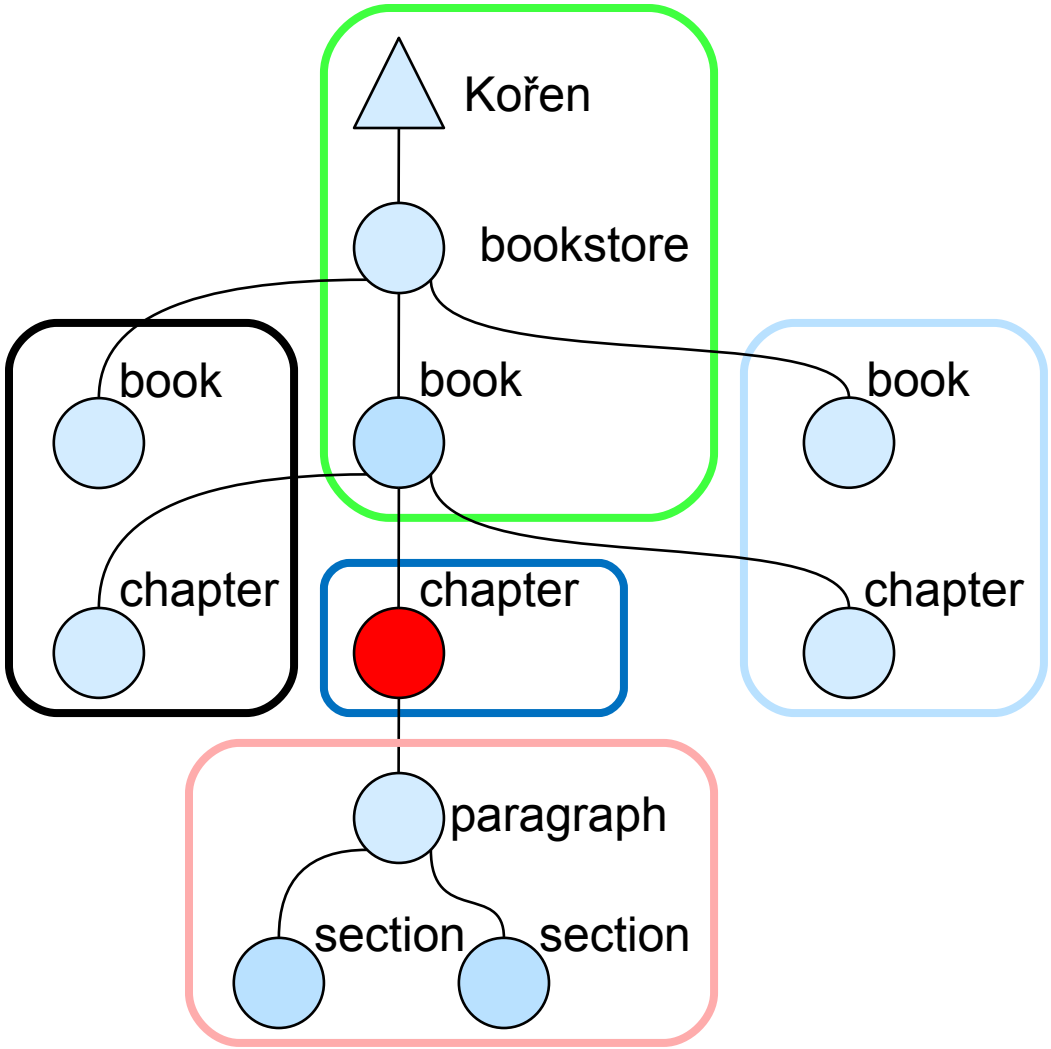


XPath - Axes - Příklad



- //chapter[2]/self::*
- //chapter[2]/preceding::*
- //chapter[2]/following::*
- //chapter[2]/ancestor::*
- //chapter[2]/descendant::*

XPath - Axes - Příklad




```
<bookstore>
  <book>
  <book>
    <chapter>
    </chapter>
    <chapter>
      <section>
        <paragraph/>
        <paragraph/>
      </section>
    </chapter>
    <chapter>
    </chapter>
  </book>
  <book/>
</bookstore>
```

XPath - Zkrácené formy zápisu axes

(nic)	je ekvivalent	child::
@	je ekvivalent	attribute::
.	je ekvivalent	self::node()
.//X	je ekvivalent	self::node()/descendant-or-self::node()/child::X
..	je ekvivalent	parent::node()
../X	je ekvivalent	parent::node()/child::X
//	je ekvivalent	/descendant-or-self::node()/
//X	je ekvivalent	/descendant-or-self::node()/child::X

Operátory

Operátor	Význam	Příklad
|	Dvě sady uzlů	//book | //cd
+	Sčítání	6 + 4
-	Odčítání	6 - 4
*	Násobení	6 * 4
div	Dělení	8 div 4
=	Rovnost	price=9.80
!=	Nerovnost	price!=9.80
<	Menší než	price<9.80
<=	Menší nebo rovno	price<=9.80
>	Větší než	price>9.80
>=	Větší nebo rovno	price>=9.80
or	nebo	price=9.80 or price=9.70
and	a	price>9.00 and price<9.90
mod	zbytek po dělení	5 mod 2

XPath funkce

- Celá řada (kategorií) funkcí
- Mají prefix fn: a URI http://www.w3.org/2005/02/xpath-functions
  - Accessor
  - Error and Trace
  - Numeric
  - String
  - AnyURI
  - Boolean
  - Duration/Date/Time
  - Qname
  - Node
  - Sequence
  - Context

Příklady XPath funkcí

Funkce	Význam
fn:node-name(node)	Jméno uzlu argumentu
fn:abs(*num*)	Absolutní hodnota argumetnu
fn:compare(*comp1,comp2*)	Porovná argumenty jako řetězce
fn:string-length()	Vrací délku řetězce aktuálního uzlu
fn:true()	Vrací hodnotu true
fn:dateTime(*date,time*)	Převede hodnoty na datumčas
fn:root(*node*)	Vrací kořenový uzel
fn:reverse(*(item1,item2,...itemN)*)	Vrací obrácené pořadí itemN,…item2,item1

Příklady XPath funkcí

```
<?xml version="1.0"
encoding="UTF-8"?>
<library>
   <book>
      <chapter/>
      <chapter>
         <section>
            <paragraph/>
            <paragraph/>
         </section>
      </chapter>
      <chapter/>
   </book>
   <book/>
</library>
```

Pozn.: Aktuálním uzlem je kořen

//chapter[count(section)=1]

Vybere chapter, které mají přesně jedno dítě section

//*[name()='section']

//section

Vybere uzel section kdekoli v dokumentu

//*[starts-with(name(), 'sec']

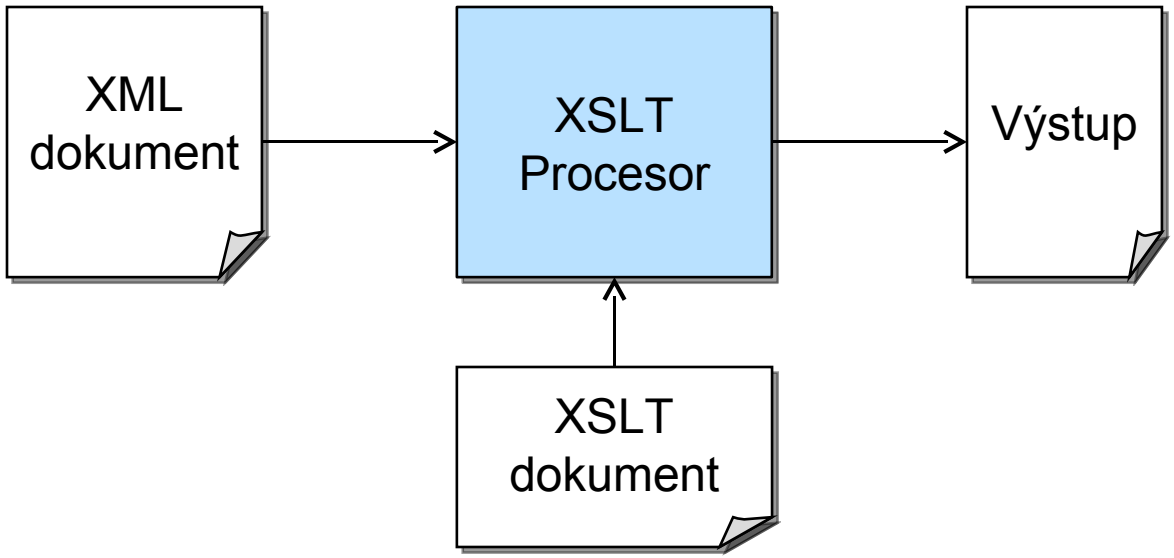
Vybere uzly, jejichž jméno začíná na 'sec' kdekoli v dokumentu

//*[contains(name(), 'ect']

Vybere uzly, jejichž jméno obsahuje řetězec 'ect' kdekoli v dokumentu

XSL Transformace (XSLT)

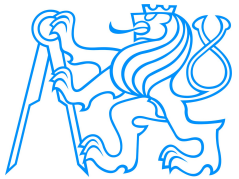
- Extensible Stylesheet Language
  - jazyk pro transformaci a formátování XML dokumentů



- Výstup může být jiný XML dokument nebo jiný typ dokumentu (např. HTML, čistý text nebo PDF)

XSLT

- Definuje sadu transformačních pravidel
- Transformační pravidlo
  - se aplikuje podle vzoru
  - definuje co se přidá na výstup
  - říká, co se bude dělat dál (např. pokračovat rekurzivně)
- XSLT Procesor začne aplikovat pravidla od kořenového uzlu
- XSLT používá stejný model XML dokumentu jako XPath



XSLT – typická struktura

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

     <!- Transofrmační pravidlo -->
     <xsl:template match="vzor">
        XSL elementy a výstup
     </xsl:template>

     <!- Další transformační pravidla -->

</xsl:stylesheet>
```

- Vzor určuje, na které uzly se bude pravidlo aplikovat
- Obsah těla pravidla, který není v XSL: jmenném prostoru, se přenese rovnou na výstup
- To, co je v XSL: jmenném prostoru se bude dále interpretovat

XSLT příklad

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

<bookstore>
    <!--Authors-->
    <authors>
        <author id="author01">
            <firstname>Josef</firstname>
            <lastname>Novak</lastname>
        </author>
        <author id="author02">
            <firstname>Jana</firstname>
            <lastname>Novotna</lastname>
        </author>
        <author id="author03">
            <firstname>Vladislav</firstname>
            <lastname>Vomacka</lastname>
        </author>
    </authors>
</bookstore>
```

XSLT – příklad cont

```
<?xml version="1.0" ?>
   <xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="html"/>

    <xsl:template match="/">
         <html>
         <head>
             <title>bookstore.xsl</title>
         </head>
         <body>
                  <xsl:apply-templates />
         </body>
         </html>
    </xsl:template>

    <xsl:template match="//author/firstname">
         <div>Jmeno=<xsl:value-of select="." /></div>
    </xsl:template>

     <xsl:template match="//author/lastname">
         <div>Prijmeni=<xsl:value-of select="." /></div>
    </xsl:template>
</xsl:stylesheet>
```

XSLT příklad výsledek

```
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>bookstore.xsl</title>
</head>
<body>
<div>Jmeno=Josef</div>
<div>Prijmeni=Novak</div>
<div>Jmeno=Jana</div>
<div>Prijmeni=Novotna</div>
<div>Jmeno=Vladislav</div>
<div>Prijmeni=Vomacka</div>
</body>
</html>
```

Provázání XSLT a XML zdroje

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

<?xml-stylesheet type="text/xsl" href="bookstore.xsl"?>

<bookstore>
    <!--Authors-->
    <authors>
        <author id="author01">
            <firstname>Josef</firstname>
            <lastname>Novak</lastname>
        </author>
        <author id="author02">
            <firstname>Jana</firstname>
            <lastname>Novotna</lastname>
        </author>
        <author id="author03">
            <firstname>Vladislav</firstname>
            <lastname>Vomacka</lastname>
        </author>
    </authors>
</bookstore>
```

Postup vykonání XSLT

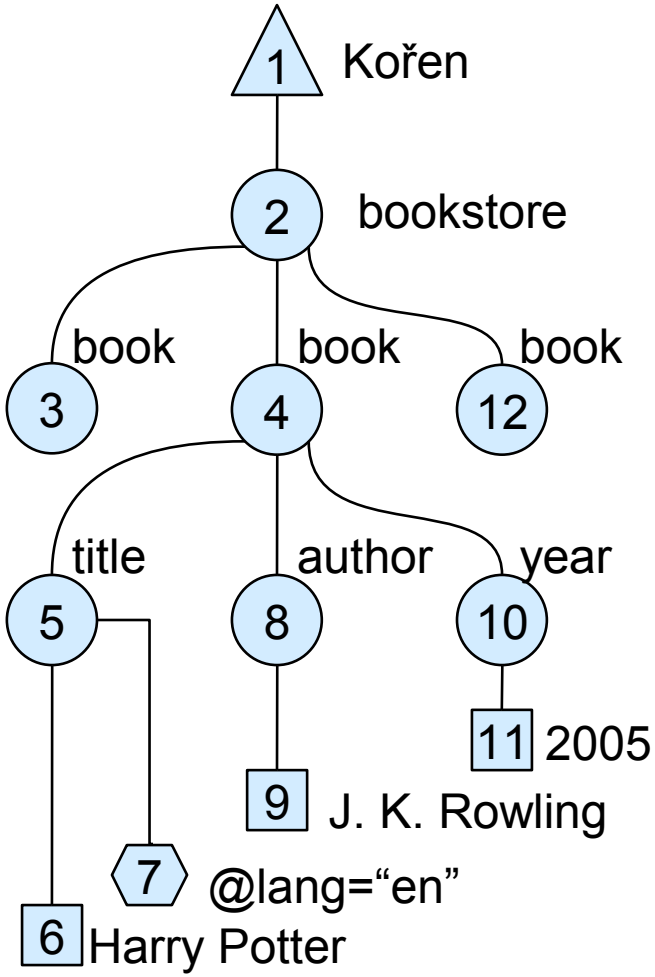
- Prochází se datový model XML dokument (od kořene)
  - Uzly jsou procházeny ve stejném pořadí v jakém jsou jejich otevírací tagy v XML dokumentu
- Nalezení všech pravidel, která se na daný uzel aplikují (XPath)
- Nalezení nejlepšího z nich
- Vykonání nejlepšího z nich

Postup při vykonání pravidla

- Při vykonávání pravidla je procházený uzel považovány za *aktuální.*
- Pravidla mohou být rekurzivní

XSLT – Způsob procházení datového modelu

```
<?xml version="1.0"
    encoding="UTF-8"?>
<bookstore>
    <book></book>
    <book>
      <title lang="en">
          Harry Potter
      </title>
      <author>J. K. Rowling</author>
      <year>2005</year>
    </book>
    <book></book>
</bookstore>
```



Nalezení nejlepšího z pravidel

- Na daný uzel se může vztahovat více pravidel

```
<xsl:template match="/authors/author">
   <!- prvni pravidlo -->
</xsl:template>

<xsl:template match="//author">
   <!- druhe pravidlo -->
</xsl:template>
```

- Lokální pravdla mají přednost před importovanými
- Pravidla jsou řazena takto od nejnižší priority

  *, foo, baz/foo, /baz/foo
- Pozor na konkrétní implementaci!

XSLT - XSL Apply Templates

- xsl:apply-templates
- Uvnitř pravidla může být výběr nové množiny uzlů
  - atribut match
  - pokud atribut match není uveden vyberou se všichni potomci aktuálního uzlu
- Vybraná množina je většinou podmnožina z potomků

  …nemusí ale být (XPath)
- Po provedení pravidla, ve kterém je xsl:apply-templates se již neprocházejí potomci aktuálního uzlu tohoto pravidla

```
<xsl:template match="//authors">

   <xsl:apply-templates match="author" />

</xsl:template>
```

XSLT – Příklad 1

Kolik je autorů?

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

<bookstore>
  <!--Authors-->
  <authors>
     <author id="author01">
        <firstname>Josef</firstname>
        <lastname>Novak</lastname>
     </author>
     <author id="author02">
        <firstname>Jana</firstname>
        <lastname>Novotna</lastname>
     </author>
     <author id="author03">

<firstname>Vladislav</firstname>
          <lastname>Vomacka</lastname>
     </author>
    </authors>
</bookstore>
```

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

  <xsl:template match="/">

   <xsl:value-of select="count(//author)"/>

  </xsl:template>

</xsl:stylesheet>
```

Výsledek

```
<?xml version="1.0" encoding="UTF-8" ?>
3
```

XSLT – Příklad 2

Vypiš všechny autory a jejich id

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

<bookstore>
  <!--Authors-->
  <authors>
     <author id="author01">
        <firstname>Josef</firstname>
        <lastname>Novak</lastname>
     </author>
     <author id="author02">
        <firstname>Jana</firstname>
        <lastname>Novotna</lastname>
     </author>
     <author id="author03">

<firstname>Vladislav</firstname>
          <lastname>Vomacka</lastname>
     </author>
    </authors>
</bookstore>
```

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

 <xsl:template match="author">
  <div>
  jmeno: <xsl:value-of select="firstname" />
  prijmeni: <xsl:value-of select="lastname"/>
  id: <xsl:value-of select="@id" />
  </div>
 </xsl:template>
</xsl:stylesheet>
```

Výsledek

```
<?xml version="1.0" encoding="UTF-8"?
<div>jmeno: Josef
     prijmeni: Novak
     id: author01</div>
<div>jmeno: Jana
     prijmeni: Novotna
     id: author02</div>
<div>jmeno: Vladislav
     prijmeni: Vomacka
     id: author03</div>
```

XSLT – Příklad 3

Vypiš příjmení autorů v jednom div elementu

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

<bookstore>
  <!--Authors-->
  <authors>
     <author id="author01">
        <firstname>Josef</firstname>
        <lastname>Novak</lastname>
     </author>
     <author id="author02">
        <firstname>Jana</firstname>
        <lastname>Novotna</lastname>
     </author>
     <author id="author03">

<firstname>Vladislav</firstname>
          <lastname>Vomacka</lastname>
     </author>
    </authors>
</bookstore>
```

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

 <xsl:template match="author">
   prijmeni: <xsl:value-of select="lastname"/>
 </xsl:template>
</xsl:stylesheet>
```

Výsledek

```
<?xml version="1.0" encoding="UTF-8"?
<div>prijmeni: Novak
     prijmeni: Novotna
     prijmeni: Vomacka</div>
```

Reference

http://www.w3.org/TR/2007/REC-xslt20-20070123/

http://www.xml.com/

http://www.biztalk.org/

http://www.xml.org/

http://www.oasis-open.org/cover/

http://zvon.vscht.cz/

http://www.xmlsoftware.com/

http://www.w3.org/XML/

http://www.wapserver.cz/