XPath

Vanessa Braganholo

Curso baseado em mini-cursos apresentados no SBBD. Autores: Carlos Heuser, Carina Dorneles e Vanessa Braganholo

XPath

- Especificação: http://www.w3.org/TR/xpath
- Uma expressão XPath seleciona um conjunto de nodos
- Operadores principais:
  - / para dar um “passo” na árvore XML (percorrer uma relação pai-filho)
  - // para dar vários “passos” de uma vez (percorrer uma relação ascendente-descendente)

Exemplo

- /empregados/empregado

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Contexto

- Cada / muda o contexto atual da consulta: /empregados

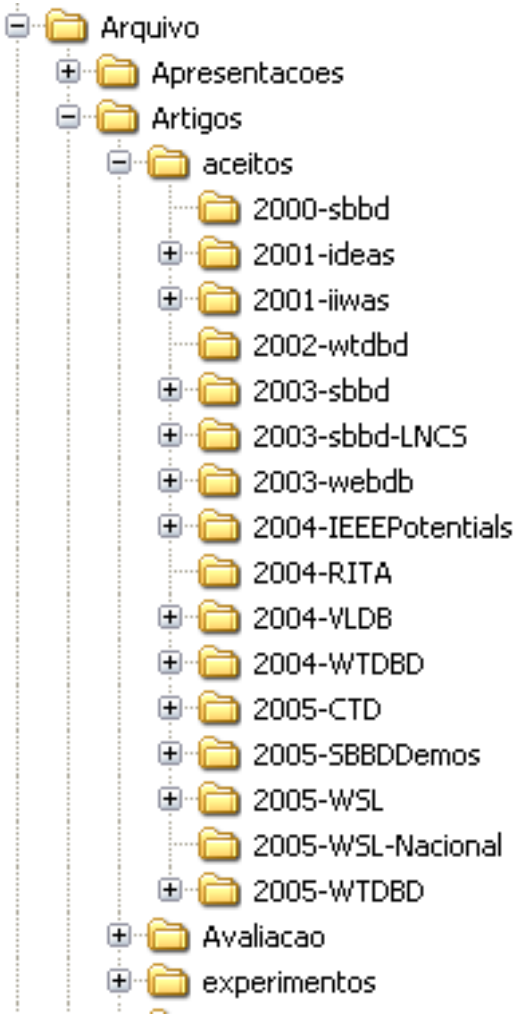
```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

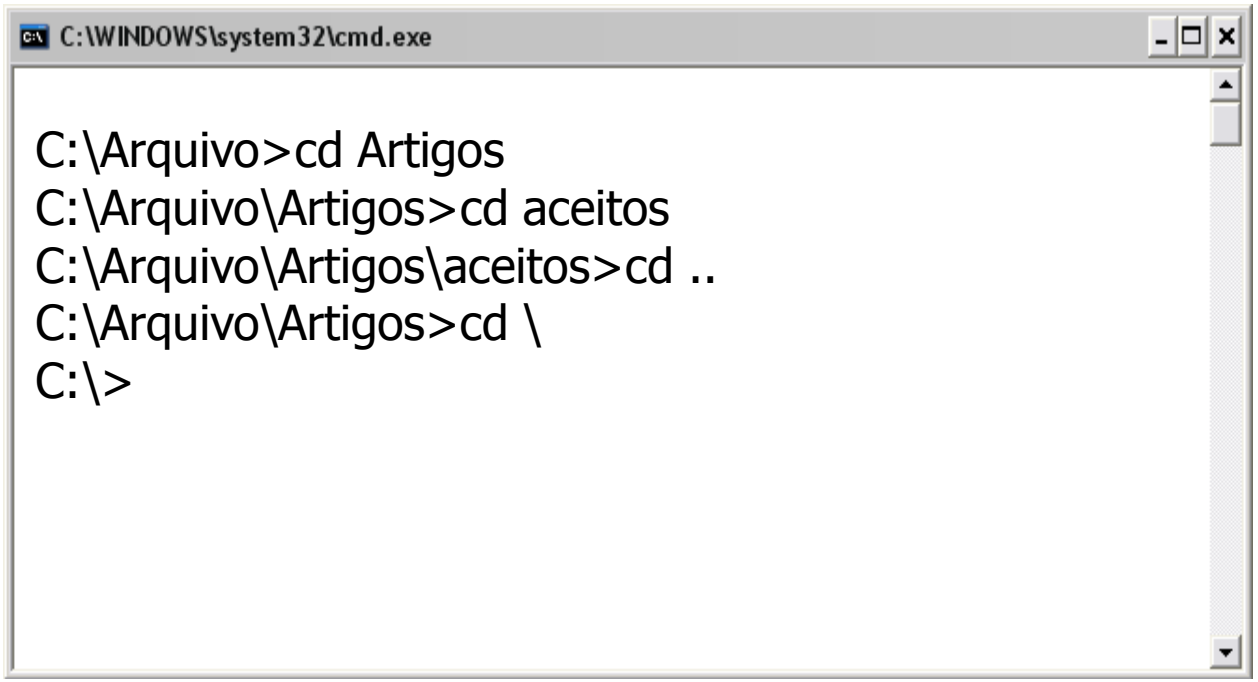
Contexto

- Cada / muda o contexto atual da consulta: /empregados/empregado

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

É como uma árvore de diretórios...





Retorno da Expressão

- Conjunto de nodos retornados é sempre o especificado pelo último passo do caminho

/empregados/empregado/nome

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
     <nome>João</nome>
     <inicial-meio>S.</inicial-meio>
     <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
     <nome>Ana</nome>
     <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Operador //

- Retorna os descendentes a partir do contexto atual

/empregados//nome //sobrenome

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
     <nome>João</nome>
     <inicial-meio>S.</inicial-meio>
     <sobrenome>Santos</sobrenome>
  </empregado>
 <empregado cod="E02" dept="D01">
     <nome>Ana</nome>
     <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
     <nome>João</nome>
     <inicial-meio>S.</inicial-meio>
     <sobrenome>Santos</sobrenome>
  </empregado>
 <empregado cod="E02" dept="D01">
     <nome>Ana</nome>
     <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Atributos

- Usa-se “@” na frente do nome do atributo
  /empregados/empregado/@cod

```
<? xml version=“1.0” ?>
<empregados>
  <empregado cod=“E01” dept=“D01”>
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod=“E02” dept=“D01”>
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Operador .

- Seleciona o elemento corrente

Equivalentes!

/empregados/.

/empregados

- Útil para uso dentro de funções

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Operador ..

- Seleciona o pai do contexto atual /empregados/empregado/..

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Operador *

- Substitui um passo do caminho

//empregado/*

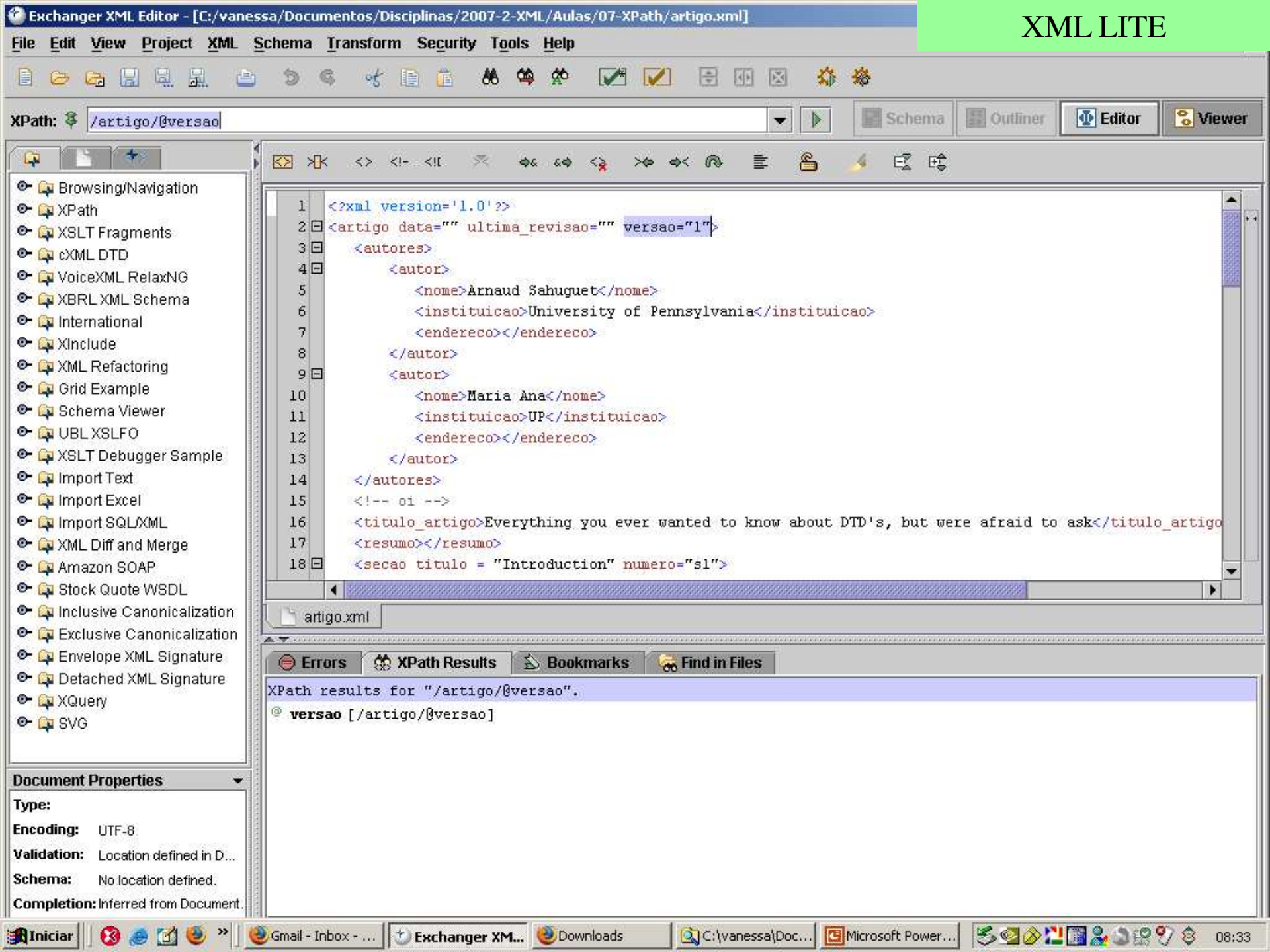
```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

XPath na prática...

- Arquivo: artigo.xml
  - Disponível na página da disciplina

- XML Exchanger Lite
  - Abrir o arquivo XML a ser consultado, digitar a expressão XPath e apertar a seta verde.

- XPath Visualizer
  - Disponível em http://www.topxml.com/xpathvisualizer/


Exchanger XML Editor - [C:/vanessa/Documentos/Disciplinas/2007-2-XML/Aulas/07-XPath/artigo.xml]
XML LITE
File Edit View Project XML Schema Transform Security Tools Help
XPath: /artigo/@versao
Schema Outliner Editor Viewer
Browsing/Navigation
XPath
XSLT Fragments
cXML DTD
VoiceXML RelaxNG
XBRL XML Schema
International
XInclude
XML Refactoring
Grid Example
Schema Viewer
UBL XSLFO
XSLT Debugger Sample
Import Text
Import Excel
Import SQL/XML
XML Diff and Merge
Amazon SOAP
Stock Quote WSDL
Inclusive Canonicalization
Exclusive Canonicalization
Envelope XML Signature
Detached XML Signature
XQuery
SVG
1 <?xml version='1.0'?>
2 <artigo data="" ultima_revisao="" versao="1">
3 <autores>
4 <autor>
5 <nome>Arnaud Sahuguet</nome>
6 <instituicao>University of Pennsylvania</instituicao>
7 <endereco></endereco>
8 </autor>
9 <autor>
10 <nome>Maria Ana</nome>
11 <instituicao>UP</instituicao>
12 <endereco></endereco>
13 </autor>
14 </autores>
15 <!-- oi -->
16 <titulo_artigo>Everything you ever wanted to know about DTD's, but were afraid to ask</titulo_artigo
17 <resumo></resumo>
18 <secao titulo = "Introduction" numero="s1">
artigo.xml
Errors XPath Results Bookmarks Find in Files
XPath results for "/artigo/@versao".
versao [/artigo/@versao]
Document Properties
Type:
Encoding: UTF-8
Validation: Location defined in D...
Schema: No location defined.
Completion: Inferred from Document.
Iniciar
Gmail - Inbox - ...
Exchanger XM...
Downloads
C:\vanessa\Doc...
Microsoft Power...
08:33

XPath Visualiser – só funciona no Internet Explorer

XPath Visualiser Ver.1.4 - Microsoft Internet Explorer

Arquivo Editar Exibir Favoritos Ferramentas Ajuda

Endereço C:\vanessa\Documentos\Disciplinas\2007-2-XML\Aulas\07-XPath\XPathVisualiser\XPathMain.htm

C:\vanessa\Documentos\Disciplinas\2007-2-XML\A Procurar... Process File

XPath expression:

/artigo/@versao Select Nodes Variables Keys

0 of 1/1 matches

```
-<artigo data="" ultima_revisao="" versao="1">
  -<autores>
    -<autor>
      <nome>Arnaud Sahuguet</nome>
      <instituicao>University of Pennsylvania</instituicao>
      <endereco />
    </autor>
    -<autor>
      <nome>Maria Ana</nome>
      <instituicao>UP</instituicao>
      <endereco />
    </autor>
  </autores>
   <!-- oi -->
  <titulo_artigo>Everything you ever wanted to know about DTD's, but were afraid to ask</titulo_artigo>
  <resumo />
  -<secao titulo="Introduction" numero="s1">
    <paragrafo>For the last TWO YEARS ... </paragrafo>
    <paragrafo>For the last two years ... </paragrafo>
    -<paragrafo>
     From a database perspective
      <citacao ref="b1" />
     for instance, DTDs can be useful to
    </paragrafo>
  </secao>
```

Concluído Meu computador

Exercício 1

- Usando o documento XML fornecido, crie expressões XPath para as seguintes consultas:

a) Selecionar as instituições dos autores do artigo
b) Selecionar todos os parágrafos das seções do artigo
c) Selecionar nomes dos autores do artigo propriamente dito e das referencias bibliográficas
d) Selecionar pai do elemento endereco
e) Selecionar avô do elemento paragrafo
f) Selecionar todas as ocorrências de endereço

Filtros

- Restringem o conjunto de nodos selecionados
- Podem ser colocados em qualquer passo do caminho

Filtros

- Sintaxe: Expressão booleana entre colchetes

  //empregado[@cod="E01"]

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
     <nome>João</nome>
     <inicial-meio>S.</inicial-meio>
     <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
     <nome>Ana</nome>
     <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Contexto do Filtro

- Sempre o último passo percorrido antes do filtro
- Retorno nunca é o que está no filtro, mas o último passo do caminho da expressão

//empregado[nome="Ana"]

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Filtro de Posição

//empregado[1]

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
     <nome>João</nome>
     <inicial-meio>S.</inicial-meio>
     <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
     <nome>Ana</nome>
     <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Filtro de Posição

//nome[1]

ATENÇÃO: o filtro de posição leva em conta o pai do nó como contexto, por isso neste exemplo ele retorna os dois elementos nome, e não apenas o primeiro que aparece no documento

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

AND, OR e NOT são aceitos

//empregado[@dept='D01' and nome='João']/sobrenome

ATENÇÃO: a ferramenta **XML Exchanger Lite** só aceita **and** e **or** minúsculos

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Testes de elementos

- O nome de um elemento pode aparecer representando um elemento que deve estar presente como um filho
  - Selecionar um elemento empregado se ele contém um sub-elemento inicial-meio: //empregado[inicial-meio]

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Exercício 2

a) Selecionar o autor cujo nome é Maria Ana
b) Selecionar a obra da bibliografia cujo ano é 1999 e o local é University of Pennsylvania
c) Selecionar a seção cujo número é s2 e que contém um parágrafo cujo conteúdo é ...
d) Selecionar o atributo título das seções

Funções

- XPath possui muitas funções
- A maioria delas é utilizada dentro dos filtros
- Uma lista completa está disponível aqui: http://www.w3.org/TR/xpath#corelib

Testes de posição

- Função position() retorna a localização sequencial do elemento testado
  - Selecionar somente o primeiro parágrafo dentre os já selecionados pelo padrão

//empregado[position()=1]

ou

//empregado[1]

Testes de posição

- Função last() localiza o último elemento (retorna o número da posição do último elemento)
  - Selecionar o último empregado
    //empregado[last()]
- Função count() retorna o número de ocorrências de um elemento
  - Selecionar seções que contenham apenas dois parágrafos
    //secao[count(paragrafo)=2]

Função NOT

- Função not() para reverter o resultado do teste
  - Selecionar todas as notas, exceto a terceira

    //nota[not(position()=3)]
  - Selecionar uma nota que não contém um elemento título

    //nota[not(titulo)]
  - Selecionar todos os capítulos, exceto aquele que tenha o atributo número com valor 10

    //capitulo[not(@numero='10')]

Comparações

- Selecionar todos paragrafos, mas não o último

  //paragrafo[position()!=last()]

- Outras comparações:

  //paragrafo[position()>2]

  //paragrafo[position()>=3]

  //paragrafo[position()>2 and position()<last()]

  //paragrafo[position()=2 or position() = 4]

Tratamento de Strings

- Função contains(par1, par2) retorna true se par1 contém o texto em par2

  par1 pode ser text() ou .

Função contains()

- Usando "text()", testa somente o conteúdo textual do elemento
  - Selecionar titulo que contenha a palavra "relacional"

```
//titulo[contains(text(), "relacional")]
```

```
<titulo>Modelo relacional</titulo>
```

Tratamento de Strings

- Usando ".", testa o elemento secao e seus subelementos
  - Selecionar secao que contenha a palavra "relacional" em seu texto ou no texto de algum de seus descendentes

```
//secao[contains(., "relacional")]
```

```
<secao>Esta secao apresenta...
   <paragrafo>O modelo relacional ...</paragrafo>
   <paragrafo>Como já mencionado, ...</paragrafo>
</secao>
```

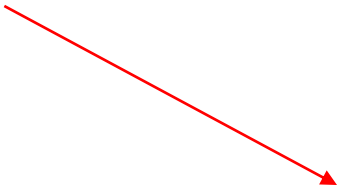
Tratamento de Strings

- Função starts-with() testa o texto no começo da string. Não pode haver espaço em branco.
  - Selecionar titulo que inicie com a palavra "Introdução"

//titulo[starts-with(., "Introdução")]

<titulo>Introdução a JSP</titulo>

<titulo>_Introdução a JSP</titulo>



Não funciona!

Tratamento de Strings

- Função string() converte o valor do argumento para string
  - Exemplo:

  string(//capitulo[1]/@numero), retorna o valor do atributo numero do primeiro capitulo, em formato string

- Função normalize-space()
  - No meio da string, reduz vários espaços em branco para um único caractere espaço
  - Remove completamente os espaços do início e fim da string

```
//titulo[contains(normalize-space(.), "Introdução a JSP")]

<titulo>  Introdução       a       JSP  </titulo>
```

Tratamento de Strings

- Função concat() concatena strings. Pode ter um ou mais parâmetros: concat(text1, texto2,..., texton)
  - Retornar a seção que fale da autora do livro

```
//secao[contains(.,concat (../autor/nome/text(), " ",
  ../autor/sobrenome/text()))]
```

```
<livro>
  <autor>
    <nome>Ana</nome>
    <sobrenome>Silva</sobrenome>
  </autor>
  <secao> A autora Ana Silva ...</secao>
  <secao> ...  </secao>
</livro>
```

Tratamento de Strings

- Função translate() converte caracteres de acordo com um esquema de mapeamento.
  - Uso: comparações case-insensitive
  - Parâmetros: string para converter, caracteres para modificar no texto fonte, e valores a serem colocados

```
//paragrafo[contains(translate(normalize-space(.),
    "abcdefghijklmnopqrstuvwxyz",
    "ABCDEFGHIJKLMNOPQRSTUVWXYZ"), "ELEMENTO")]
```

```
<paragrafo>EM XML, UM ELEMENTO É ...</paragrafo>
<paragrafo>Um documento XML deve possuir um elemento raiz
    ...</paragrafo>
```

  - os dois elementos paragrafo são recuperados

Tratamento de Números

- Operadores + e - podem ser usados:
  - //nota[3] é equivalente a //nota[1 + 2]

- Função mod()
  - Fornece o resto de uma divisão truncada
  - Selecionar parágrafos pares:

    //paragrafo[position() mod 2 = 0]

Filtros Múltiplos

- Usados para combinar um teste de posição e um outro tipo de teste
  - Selecionar nomes de companhias, e então extrair o terceiro nome da lista //nome[companhia][3]

```
...
<nomes>
  <nome><pessoa>...</pessoa></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><pessoa>...</pessoa></nome>
</nomes>
<nomes>
  <nome><pessoa>...</pessoa></nome>
  <nome><companhia>...</companhia></nome>
  <nome><pessoa>... </pessoa></nome>
</nomes>
...
```

Filtros Múltiplos

- Usados para combinar um teste de posição e um outro tipo de teste
  - Selecionar o terceiro nome, fornecendo o nome da companhia (só seleciona se for uma companhia!)

    //nome[3][companhia]

```
...
<nomes>
  <nome><pessoa>...</pessoa></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><pessoa>...</pessoa></nome>
</nomes>
<nomes>
  <nome><pessoa>...</pessoa></nome>
  <nome><companhia>...</companhia></nome>
  <nome><companhia>...</companhia></nome>
  <nome><pessoa>... </pessoa></nome>
</nomes>
...
```

Outras funções

- Comentários:
  - Selecione comentários que estejam dentro de livro

    //livro/comment()

- Instruções de processamento
  - Encontre instruções de processamento que estejam dentro do elemento livro

    //livro/processing-instruction()

Exercício 3

- Usando o mesmo documento XML, crie expressões XPath (pode ser necessário usar o resumo das funções XPath que está nos próximos slides)
  a) Retorne todas as seções do artigo que possuem pelo menos um subelemento figura
  b) Selecione as seções ímpares
  c) Selecione as seções ímpares que possuem pelo menos um subelemento figura
  d) Selecione os parágrafos que contêm uma instrução de processamento
  e) Encontre autores que possuam "Ana" no nome. Não devem estar nas referencias bibliográficas.
  f) A versão do artigo
  g) Selecione o parágrafo que tenha ambos os atributos "numero" e "tipo"
  h) Nome do(s) autor(res) da bibliografia cujo titulo da obra é "Union Types for Semistructured Data ".
  i) Selecione o ano da publicação de bibliografias, cujo nome do autor contenha a palavra "Abiteboul"

Exercício 3

j) Um parágrafo cujo tamanho da string que ele contém é 27
k) Selecione o terceiro parágrafo de uma seção
l) Selecione um parágrafo que contenha o nome do segundo autor do artigo e sua instituição. Entre estes dois dados existe a string "from“
m) Selecione todo e qualquer parágrafo que tenha a string "two years". Trate o uso de maiúsculas e minúsculas!!
n) Selecionar seções que contenham somente dois parágrafos
o) Selecione os elementos "ano" descendentes de bibliografia
p) Encontre a instrução de processamento de algum elemento paragrafo.

Resumo das funções, XPath 1.0

Tabela baseada nas funções apresentadas na página da W3C

- Funções para *nodos (elementos)*

Nome	***Sintaxe***	***Descrição***
`count()`	`count(node-set) = number`	Retorna o número de nodos de um node-set
`id()`	`id(value) = node-set`	Seleciona elementos pelo seu ID único
`last()`	`last() = number`	Retorna o número da posição do ultimo nodo em uma lista de nodos processados
`local-name()`	`local-name(node)= string`	Retorna a parte local de um nodo. Um nodo geralmente consiste de um prefixo, uma vírgula e seguida de um nome local
`name()`	`name(node) = string`	Retorna o nome de um nodo
`namespace-uri()`	`namespace-uri(node) = uri`	Retorna a URI da *namespace* de um nodo específico
`position()`	`position() = number`	Retorna a posição em uma lista de nodos do nodo que está sendo processado

Resumo das funções, XPath 1.0

Tabela baseada nas funções apresentadas na página da W3C

- Funções para *string*

Nome	***Sintaxe e Exemplo***	***Descrição***

`concat()`	`string=concat(val1, val2, ..)`
	Exemplo:
	`concat('The',' ','XML')`
	`Resultado: 'The XML'`
		Retorna a concatenação de todos os seus argumentos

`contains()`	`bool=contains(val,substr)`
	Exemplo:
	`contains('XML','X')`
	`Resultado: true`
		Retorna true se a segunda string está contida na primeira

`normalize-space()`	`string=normalize-space(string)`
	Exemplo:
	`normalize-space(' The XML ')`
	`Resultado: 'The XML'`
		Normaliza os espaços em broncos para um só

`starts-with()`	`bool=starts-with(string,substr)`
	Exemplo:
	`starts-with('XML','X')`
	`Resultado: true`
		Retorna true se a primeira string inicia com a segunda

`string()`	`string(value)`
	Exemplo:
	`string(314)`
	`Resultado: '314'`
		Converte o valor do argumento para string

Resumo das funções, XPath 1.0

Tabela baseada nas funções apresentadas na página da W3C

- Funções para *string*

Nome	***Sintaxe e Exemplo***	***Descrição***

`string-length()`	`number=string-length(string)`
	`Exemplo:`
	`string-length('Beatles')`
	`Resultado: 7`
		Retorna o número de caracteres em uma string

`substring()`	`string=substring(string,start,length)`
	`Exemplo:`
	`substring('Beatles',1,4)`
	`Resultado: 'Beat'`
		Retorna a parted a string indicada nos argumentos

`substring-after()`	`string=substring-after(string,substr)`
	`Exemplo:`
	`substring-after('12/10','/')`
	`Resultado: '10'`
		Retorna a parte da string que está depois do argumento substr

`substring-before()`	`string=substring-before(string,substr)`
	`Exemplo:`
	`substring-before('12/10','/')`
	`Resultado: '12'`
		Retorna a parted a string que está antes do argumento substr

`translate()`	`string=translate(value,string1,string2)`
	`Exemplo:`
	`translate('12:30','30','45')`
	`Resultado: '12:45'`
	`translate('12:30','03','54')`
	`Resultado: '12:45'`
	`translate('12:30','0123','abcd')`
	`Resultado: 'bc:da'`
		Executa reposição character a character.

Resumo das funções, XPath 1.0

Tabela baseada nas funções apresentadas na página da W3C

- Funções para *numéricos*

Nome	***Sintaxe e Exemplo***	***Descrição***

`ceiling()`	`ceiling(number) = number`
	`Exemplo:`
	`ceiling(3.14)`
	`Resultado: 4`
		Retorna o menor inteiro que não pe menor do que o argumento

`floor()`	`floor(number) = number`
	`Exemplo:`
	`floor(3.14)`
	`Resultado: 3`
		Retorna o maior inteiro que não é maior do que o argumento

`number()`	`number(value) = number`
	`Exemplo:`
	`number('100')`
	`Resultado: 100`
		Converte o valor do argumento para um numérico

`round()`	`round(number)= integer`
	`Exemplo:`
	`round(3.14)`
	`Resultado: 3`
		Arredonda o argumento ao inteiro mais próximo

`sum()`	`sum(nodeset)=number`
	`Exemplo:`
	`sum(/cd/price)`
		Retorna o valor total de um conjunto numérico de valores em um node-set

Resumo das funções, XPath 1.0

Tabela baseada nas funções apresentadas na página da W3C

- Funções *booleanas*

Nome	***Sintaxe e Exemplo***	***Descrição***
`boolean()`	`bool=boolean(value)`	Converte o argumento e retorna true ou false

`false()`	`false()`
	
	`Exemplo:`
	`number(false())`
	`Resultado: 0`
		Retorna false

`lang()`	`bool=lang(language)`	Retorna true se a linguagem do argumento casa com a linguagem do elemento xsl:lang

`not()`	`bool=not(condition)`
	
	`Exemplo:`
	`not(false())`
		Retorna true se a condição de argumento for falsa, e falsa se a condição for verdadeira

`true()`	`true()`
	
	`Exemplo:`
	`number(true())`
	`Resultado: 1`
		Retorna true

XPath e Java

- Java tem um pacote para lidar com expressões XPath
  - **javax.xml.xpath**
  - Documentação: http://java.sun.com/j2se/1.5.0/docs/api/javax/xml/xpath/package-summary.html