XPath
lokalizácia v strome

XPath

nástroj na adresáciu jednotlivých častí XML dokumentu

zdieľaný medzi viacerými XML jazykmi (XSchema, XSLT, …)

umožňuje:
- výber dát z XML dokumentu
- porovnávanie dát (matching) so vzormi (pattern)
- základnú manipuláciu s reťazcami, číslami, bool…

XPath (pokr.) (príklady)

neXML syntax

plná podpora **namespace**

Jednoduché príklady:

```
child::para
descendant::figure[position()=3]
```


Reprezentácia XML dokumentu

XPath pracuje nad abstraktnou reprezentáciou XML dokumentu - **strom**

uzly (nodes):
- koreň
- element
- text
- atribút
- namespace
- processing instruction
- komentár

Reprezentácia XML dokumentu (pokr.)

každý uzol (okrem koreňa) má práve jedného rodiča

pre každý uzol existuje textová reprezentácia

definované poradie uzlov: **document order**
podľa poradia začiatočných značiek v XML dokumente
určuje napr. poradie detí

Štruktúry jazyka XPath

výraz (expression)

vyhodnocovaný, výsledok je niektorý dátový typ

štyri základné **dátové typy**:
- **node-set**, množina uzlov
- **boolean**
- **číslo** (floating point)
- **reťazec** (postuposť znakov)

Vyhodnocovanie výrazu

v **kontexte** (aktuálneho) uzla
- relatívna pozícia (rodič, predchodca, deti, …)
- namespace (default, dedený, …)

- množina definovaných premenných
- množina (pred)definovaných funkcií

Location path

najčastejšie používaný výraz

```
child::*
child::text()
attribute::name
descendant::div
ancestor-or-self::div/child::para
/
/descendant::para[attribute::who="me"]
```

Poznámka: kompletné príklady neskôr

štruktúrované do krokov, oddelené ‘ / ‘

Location step

tri časti:

os (axis)
vzťah kontextového uzla k vyberaným v strome

test uzla
určuje typ a (úplné) meno vyberaných uzlov

predikát (nula a viac)
bližšie určuje výberové kritériá

```
child::chapter/descendant::para
following-sibling::para[position()=1]
child::*[self::chapter][attribute::who="Jr."]
```


Definované osi v strome

• **child**	deti kontextového uzla
• **descendant**	potomkovia kontextového uzla
• **parent**	rodič kontextového uzla
• **ancestor**	predchodca kontextového uzla
• **following-sibling**	nasledujúci súrodenci kontextového uzla
• **preceding-sibling**	predchádzajúci súrodenci kontextového uzla
• **following**	nasledujúce uzly (*document order*) kontex. uzla
• **preceding**	predchádzajúce (*document order*) kontex. Uzla (okrem potomkov / predchodcov)
• **attribute**	atribúty uzla
• **namespace**	namespace kontext
• **self**	samotný kontextový uzol
• **descendant-or-self**	potomok alebo samotný kontextový uzol
• **ancestor-or-self**	predchodca alebo samotný kontextový uzol

Test uzla

test **typu** uzla a (úplného) **mena** uzla

typy uzlov: atribút, namespace a element

‘ * ‘ vyberá všetky mená (pre uzly daného typu)

testovanie typu:

text()
comment()
processing-instruction(*[meno]***)**
node()

Predikáty

filtrujú množinu uzlov (výsledok výberu)

uzly číslované od 1

“vzdialenosť” od kontextového uzla (document order)

výsledkom je **nová množina** uzlov

vyhodnotenie výrazu pre každý uzol

Location path - ďalšie príklady

```
attribute::*
child::*/child::para
self::div
/descendant::olist/child::item
child::para[position()=last()-1]
following-sibling::chapter[position()=1]
child::*[self::chapter or self::appendix]
child::para[position()=5][attribute::type="a"]
```

Poznámka: kompletné príklady neskôr

značne “rozvláčny” zápis - skratky?

Skrátený zápis

- os ‘ **child::** ‘ nie je treba písať
- os ‘ **attribute::** ‘ možno skrátiť na ‘ @ ‘
- ‘ // ‘ je skrátený zápis pre ‘ /**descendant-or-self::node()**/ ‘

```
     div//para
child::div/descendant-or-self::node()/child::para
```

- ‘ **.** ‘ je skrátený zápis pre ‘ **self::node()** ‘ = kontext. uzol

```
.//para
self::node()/descendant-or-self::node()/child::para
```

- ‘ **..** ’ je skrátenie pre ‘ **parent::node()** ’

Skrátený zápis - príklady

```
para
*
text()
@name
@*
para[last()]
para[@type="a"]
*/para
/doc/chapter[5]/selection[2]
```

Poznámka: kompletné príklady neskôr

Skrátený zápis - príklady

```
chapter//para
//para
//olist/item
.//para
../@lang
chapter[title="XPath"]
para[@type="a"][5]
para[5][@type="a"]
employee[@secretary and @assistant]
```

Poznámka: kompletné príklady neskôr

XPath - príklady

XML dokument - **strom** s rôznymi **osami** pohybu

výraz (expression)

- location path

dátové typy (množina uzlov, reťazec, číslo, boolean)

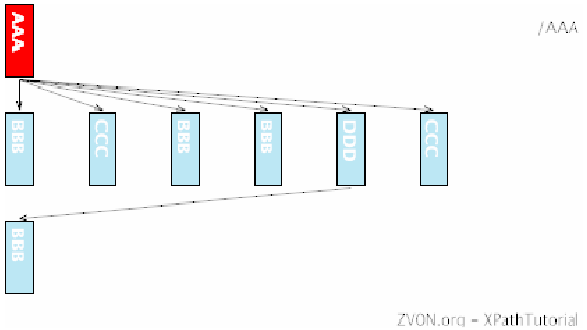
funkcie (preddefinovaná sada)

sada príkladov, prebraté z **www.zvon.org**

XPath - príklady www.zvon.org

/AAA

vyber koreňový element AAA

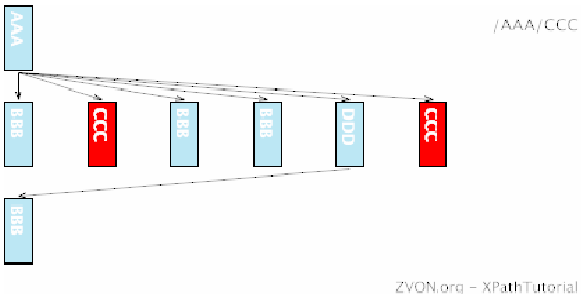



```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/CCC

vyber všetky elementy CCC, ktoré sú dieťaťom koreňového elementu AAA

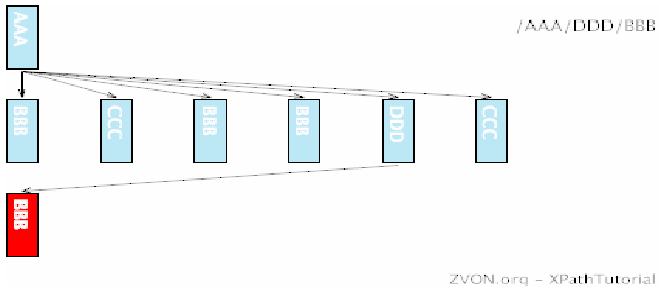



```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/DDD/BBB

vyber všetky elementy BBB, ktoré sú dieťaťom elementu DDD, ktorý je dieťaťom koreňového elementu AAA

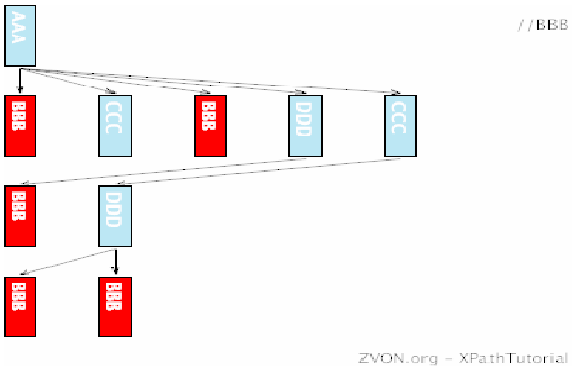



```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```

XPath - príklady www.zvon.org

//BBB

vyber všetky elementy BBB

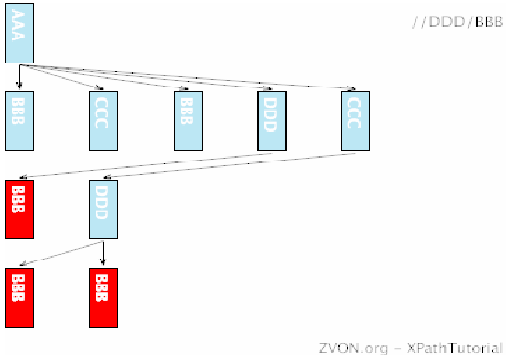



```
<AAA>
  <BBB/>
  <CCC/>
  <BBB/>
  <DDD>
    <BBB/>
  </DDD>
  <CCC>
    <DDD>
      <BBB/>
      <BBB/>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

//DDD/BBB

vyber všetky elementy BBB, ktoré sú dieťaťom elementu DDD

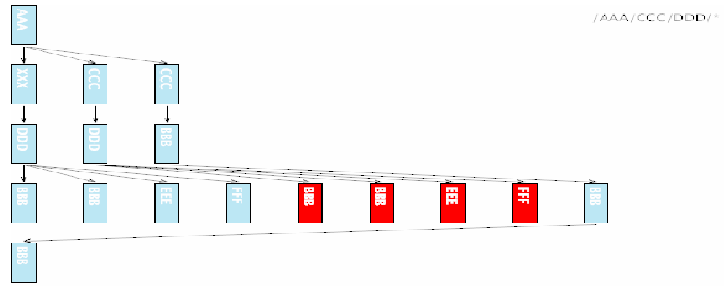



```
<AAA>
  <BBB/>
  <CCC/>
  <BBB/>
  <DDD>
    <BBB/>
  </DDD>
  <CCC>
    <DDD>
      <BBB/>
      <BBB/>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/CCC/DDD/*

vyber všetky elementy, ktoré majú rodiča DDD, ktorý má rodiča CCC, ktorý má rodiča koreňový element AAA

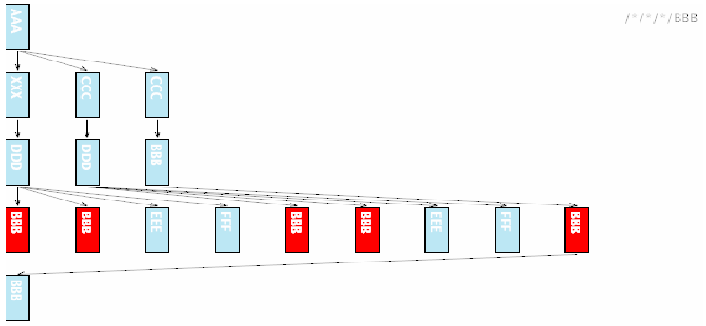



```
<AAA>
  <XXX>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </XXX>
  <CCC>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </CCC>
  <CCC>
    <BBB>
      <BBB>
        <BBB/>
      </BBB>
    </BBB>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

/*/*/*/BBB

vyber všetky elementy BBB, ktoré majú troch predkov

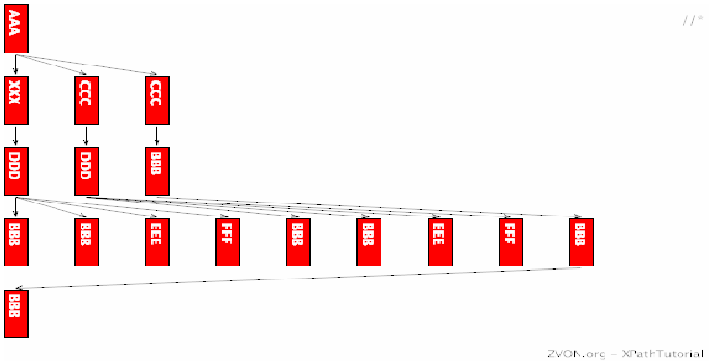



```
<AAA>
  <XXX>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </XXX>
  <CCC>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </CCC>
  <CCC>
    <BBB>
      <BBB>
        <BBB/>
      </BBB>
    </BBB>
  </CCC>
</AAA>
```

XPath - príklady www.zvon.org

//*

vyber všetky elementy

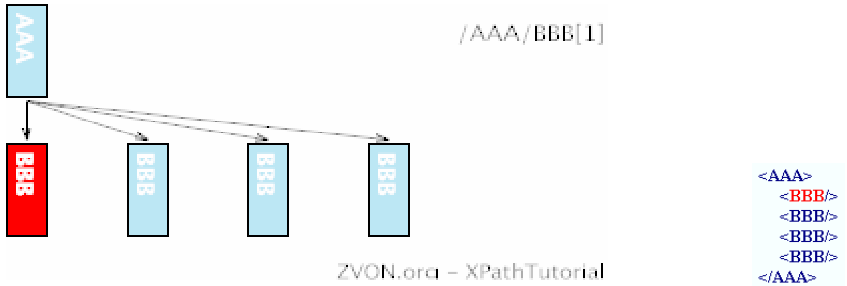



```
<AAA>
  <XXX>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </XXX>
  <CCC>
    <DDD>
      <BBB/>
      <BBB/>
      <EEE/>
      <FFF/>
    </DDD>
  </CCC>
  <CCC>
    <BBB>
      <BBB>
        <BBB/>
      </BBB>
    </BBB>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/BBB[1]

vyber prvý element BBB, ktorý je dieťaťom koreňového elementu AAA

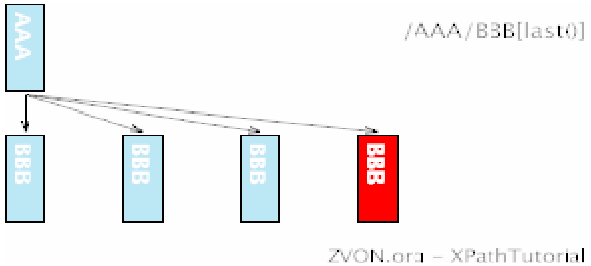



```
<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/BBB[last()]

vyber posledný element BBB, ktorý je dieťaťom koreňového elementu AAA

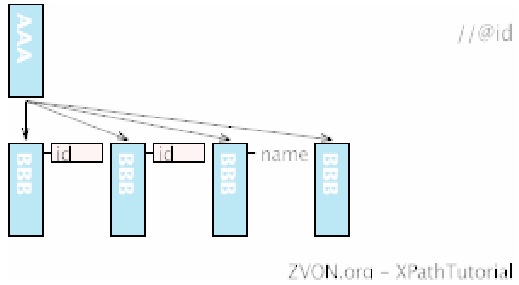



```
<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
</AAA>
```


XPath - príklady www.zvon.org

//@id

vyber všetky atribúty s menom id

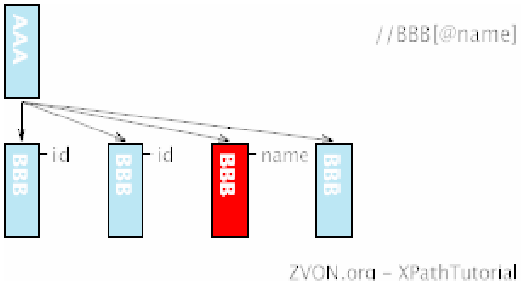



```
<AAA>
  <BBB id = "b1"/>
  <BBB id = "b2"/>
  <BBB name = "bbb"/>
  <BBB/>
</AAA>
```

XPath - príklady www.zvon.org

//BBB[@name]

vyber všetky elementy BBB, ktoré majú atribút name

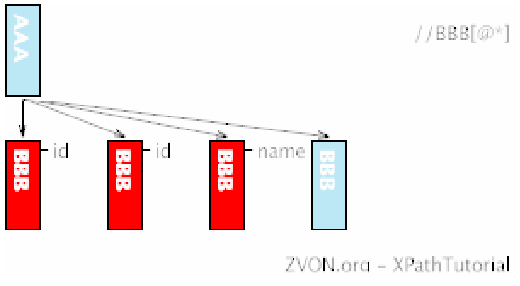



```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```


XPath - príklady www.zvon.org

//BBB[@*]

vyber všetky elementy BBB, ktoré majú nejaký atribút

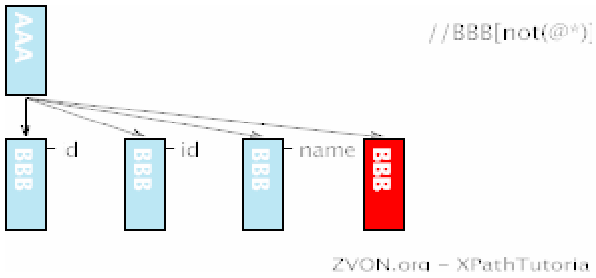



```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```


XPath - príklady www.zvon.org

//BBB[not(@*)]

vyber všetky elementy BBB, ktoré nemajú žiaden atribút

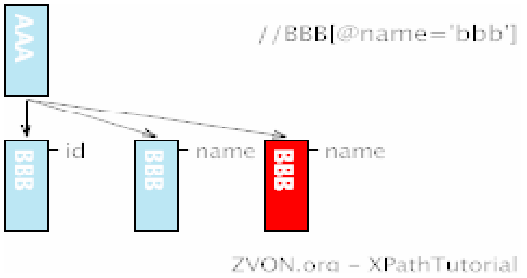



```
<AAA>
   <BBB id = "b1"/>
   <BBB id = "b2"/>
   <BBB name = "bbb"/>
   <BBB/>
</AAA>
```


XPath - príklady www.zvon.org

//BBB[@name='bbb']

vyber všetky BBB, ktorých atribút name má hodnotu “bbb”

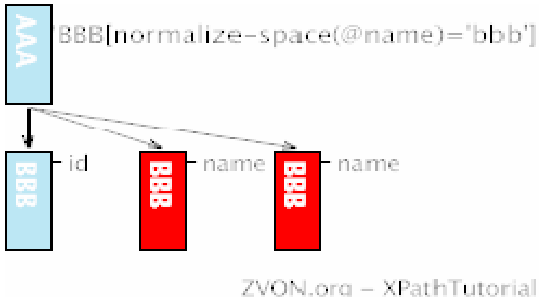



```
<AAA>
   <BBB id = "b1"/>
   <BBB name = " bbb "/>
   <BBB name = "bbb"/>
</AAA>
```

XPath - príklady www.zvon.org

//**BBB[normalize-space(@name)='bbb']**

vyber všetky BBB, ktorých atribút name má normalizovanú hodnotu “bbb”

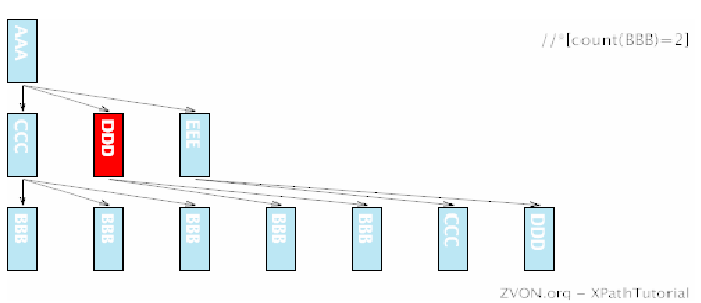



```
<AAA>
   <BBB id = "b1"/>
   <BBB name = " bbb "/>
   <BBB name = "bbb"/>
</AAA>
```


XPath - príklady www.zvon.org

//***[count(BBB)=2]**

vyber všetky elementy, ktoré majú dve deti BBB

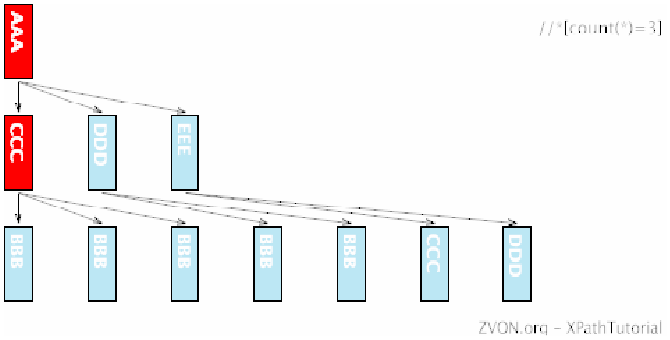



```
<AAA>
   <CCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </CCC>
   <DDD>
      <BBB/>
      <BBB/>
   </DDD>
   <EEE>
      <CCC/>
      <DDD/>
   </EEE>
</AAA>
```


XPath - príklady www.zvon.org

//***[count(*)=3]**

vyber všetky elementy, ktoré majú tri deti

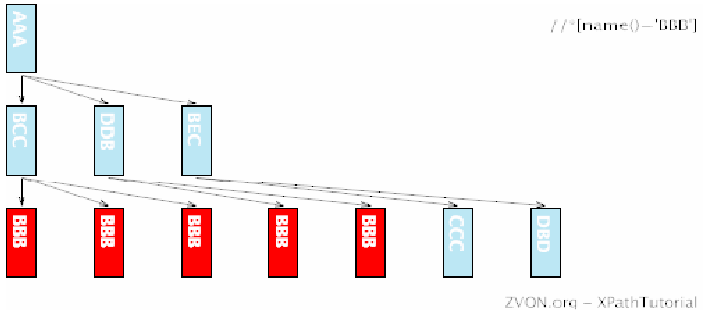



```
<AAA>
   <CCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </CCC>
   <DDD>
      <BBB/>
      <BBB/>
   </DDD>
   <EEE>
      <CCC/>
      <DDD/>
   </EEE>
</AAA>
```


XPath - príklady www.zvon.org

//***[name()=‘BBB’]**

vyber všetky elementy, ktorých meno je “BBB” (výsledok totožný s //**BBB**)

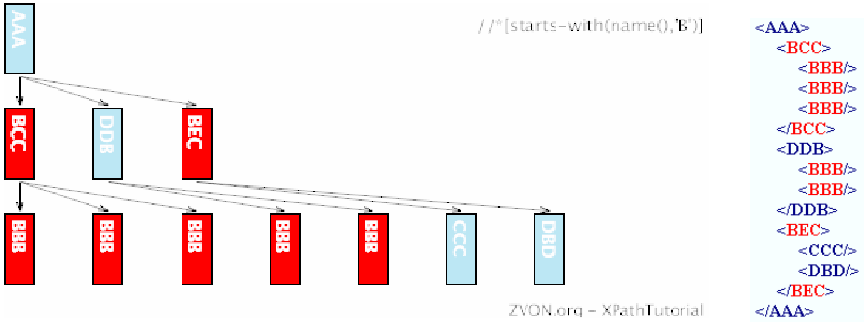



```
<AAA>
   <BCC>
      <BBB/>
      <BBB/>
      <BBB/>
   </BCC>
   <DDB>
      <BBB/>
      <BBB/>
   </DDB>
   <BEC>
      <CCC/>
      <DBD/>
   </BEC>
</AAA>
```

XPath - príklady www.zvon.org

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

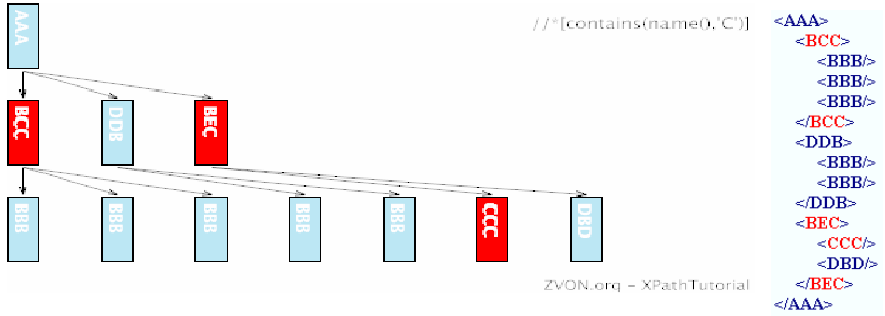
vyber všetky elementy, ktorých meno začína na "B"



XPath - príklady www.zvon.org

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

vyber všetky elementy, ktorých meno obsahuje "C"

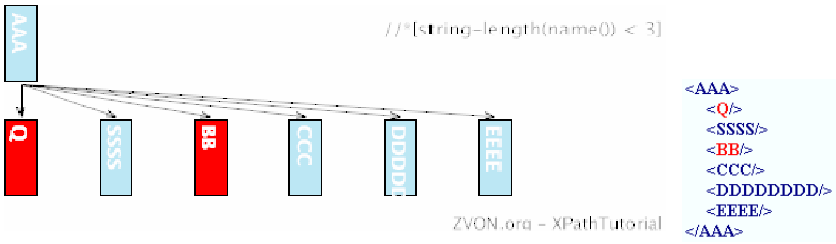


XPath - príklady www.zvon.org

//*[string-length(name()) < 3]

vyber všetky elementy, ktorých meno je kratšie ako tri znaky

Poznámka: v XML musí byť zapísané ako <

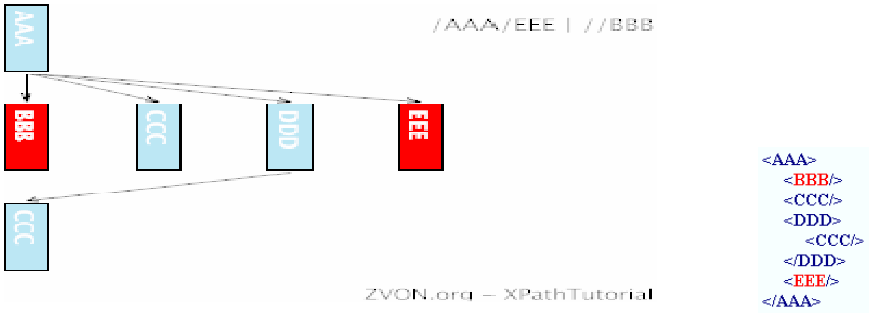


XPath - príklady www.zvon.org

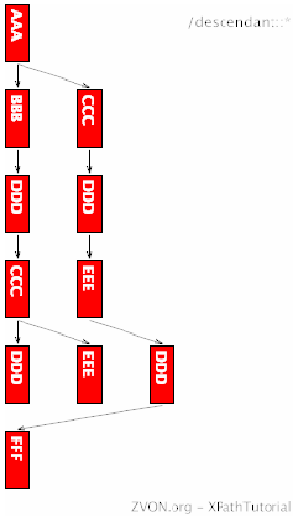
/AAA/EEE or //BBB

vyber všetky elementy, ktoré spĺňajú /AAA/EEE alebo //BBB

Poznámka: zápis aj ' | '



XPath - príklady www.zvon.org

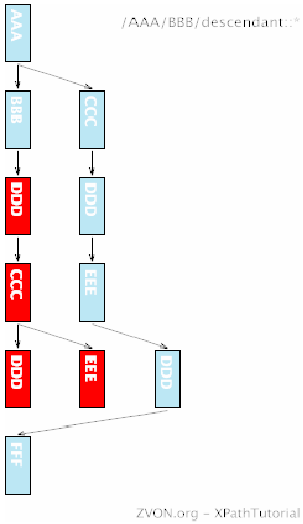


/descendant::*

vyber všetkých potomkov koreňového uzla - teda všetky uzly

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org



/AAA/BBB/descendant::*

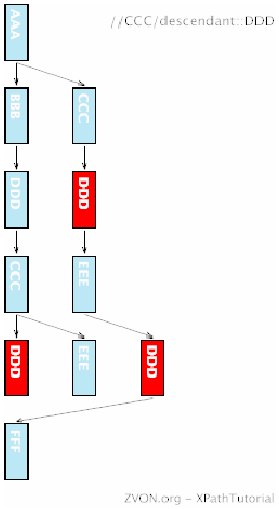
vyber všetkých potomkov pre /AAA/BBB

descendant-or-self

pribudne self

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

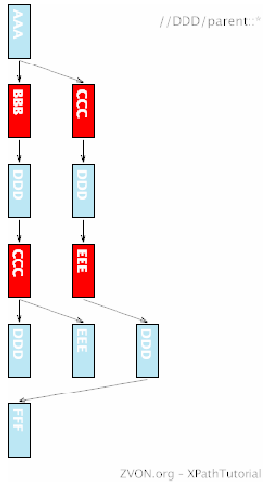


//CCC/descendant::DDD

vyber všetky elementy DDD, ktoré sú potomkom elementu CCC

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

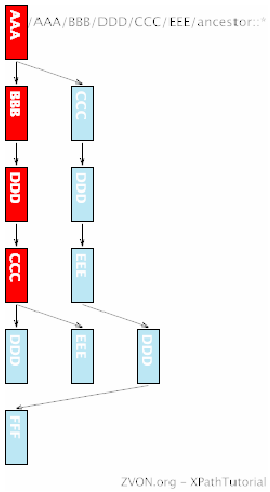


//DDD/parent::*

vyber všetkých rodičov elementov DDD

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```

XPath - príklady www.zvon.org



/AAA/BBB/DDD/CCC/EEE/ancestor::*

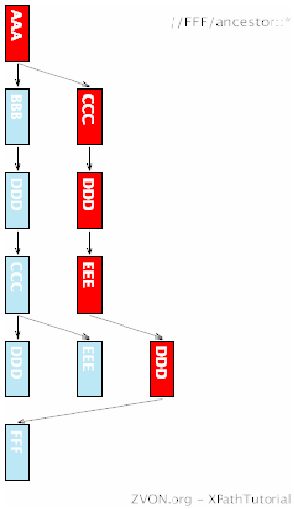
vyber predchodcov pre /AAA/...

reverse document order

ancestor-or-self

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org



//FFF/ancestor::*

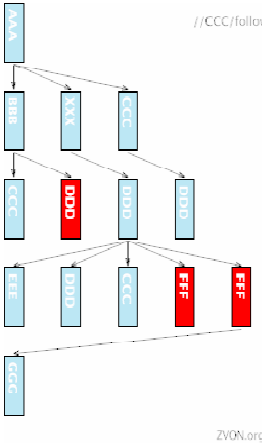
vyber predchodcov elementov FFF

```
<AAA>
  <BBB>
    <DDD>
      <CCC>
        <DDD/>
        <EEE/>
      </CCC>
    </DDD>
  </BBB>
  <CCC>
    <DDD>
      <EEE>
        <DDD>
          <FFF/>
        </DDD>
      </EEE>
    </DDD>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

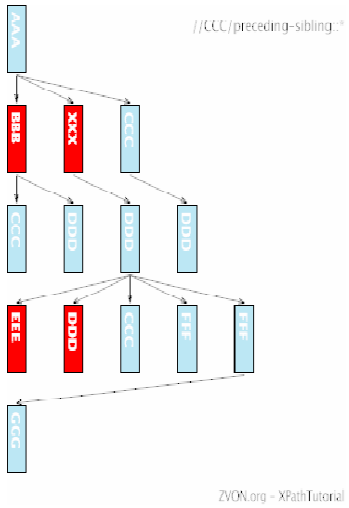
//CCC/following-sibling::*

vyber nasledujúcich súrodencov elementov CCC




```
<AAA>
  <BBB>
    <CCC/>
    <DDD/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org



//CCC/preceding-sibling::*

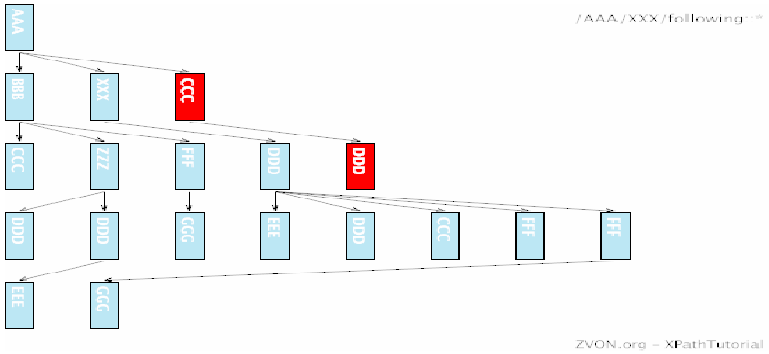
vyber predchádzajúce sesterské elementy k elementom CCC

```
<AAA>
  <BBB>
    <CCC/>
    <DDD/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

XPath - príklady www.zvon.org

/AAA/XXX/following::*

vyber všetky elementy za /AAA/XXX podľa document order, vylúč potomkov (a atrib., namesp.)

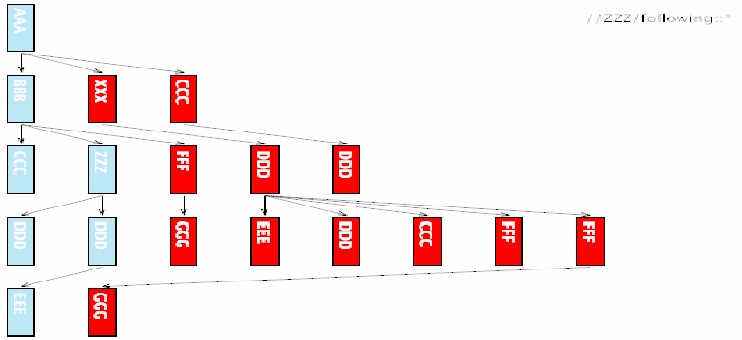



```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ>
      <DDD/>
      <DDD>
        <EEE/>
      </DDD>
    </ZZZ>
    <FFF>
      <GGG/>
    </FFF>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

//ZZZ/following::*

vyber podľa document order nasledovníkov elementov ZZZ (vylúč potomkov)

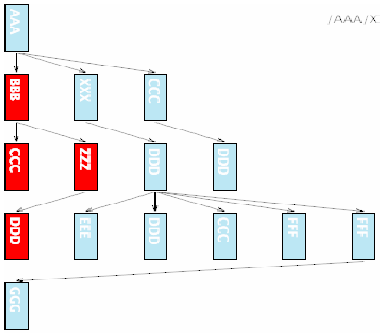



```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ>
      <DDD/>
      <DDD>
        <EEE/>
      </DDD>
    </ZZZ>
    <FFF>
      <GGG/>
    </FFF>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

/AAA/XXX/preceding::*

vyber predchádzajúce v document order, vylúč prechodcov (a attr., namesp.)

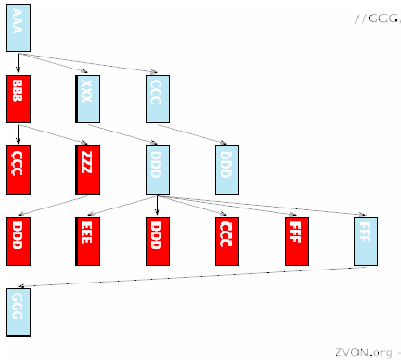



```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ>
      <DDD/>
    </ZZZ>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org

//GGG/preceding::*

vyber všetky elementy predchádzjúce GGG podľa document order (vylúč predchodcov)

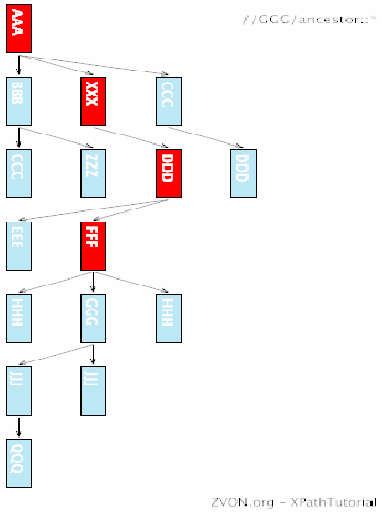



```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ>
      <DDD/>
    </ZZZ>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <DDD/>
      <CCC/>
      <FFF/>
      <FFF>
        <GGG/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

XPath - príklady www.zvon.org

osi **ancestor**, **descendant**, **following**, **preceding** a **self** sa navzájom dopĺňajú, v žiadnom elemente sa neprekrývajú a spolu pokrývajú celý strom XML dokumentu

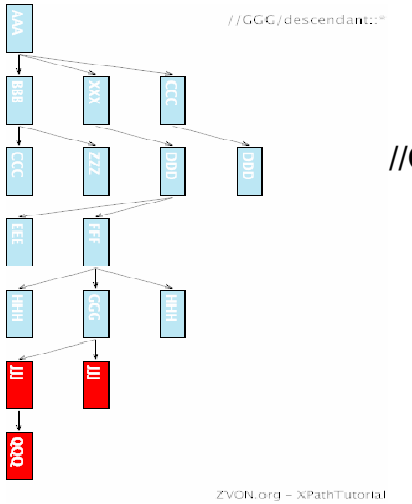
XPath - príklady www.zvon.org



//GGG/ancestor::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

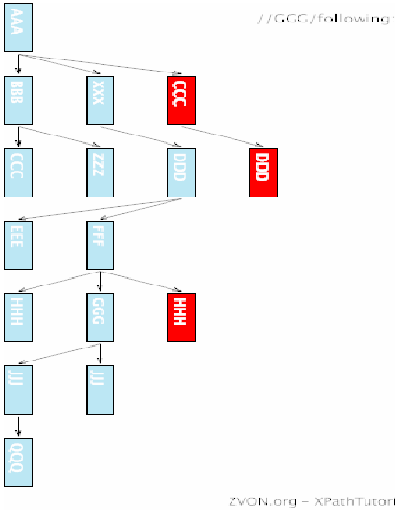

XPath - príklady www.zvon.org



//GGG/descendant::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

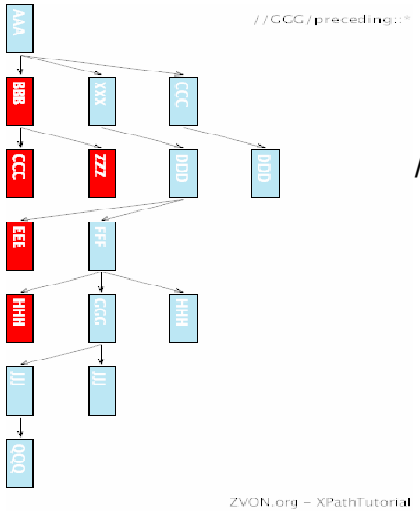

XPath - príklady www.zvon.org



//GGG/following::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

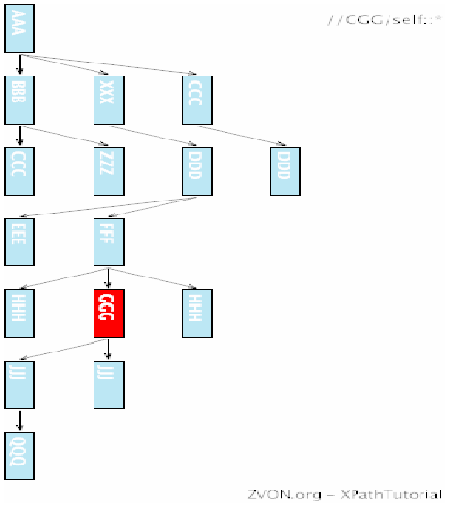
XPath - príklady www.zvon.org



//GGG/preceding::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

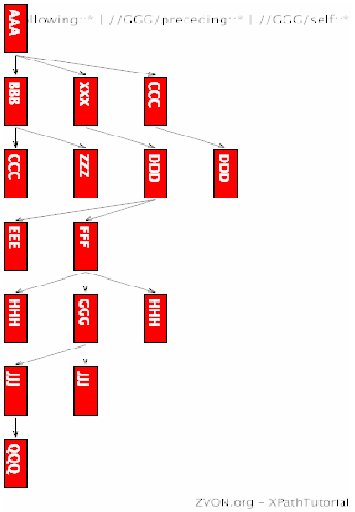

XPath - príklady www.zvon.org



//GGG/self::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```


XPath - príklady www.zvon.org



//GGG/ancestor::* |
//GGG/descendant::* |
//GGG/following::* |
//GGG/preceding::* |
//GGG/self::*

```
<AAA>
  <BBB>
    <CCC/>
    <ZZZ/>
  </BBB>
  <XXX>
    <DDD>
      <EEE/>
      <FFF>
        <HHH/>
        <GGG>
          <JJJ>
            <QQQ/>
          </JJJ>
          <JJJ/>
        </GGG>
        <HHH/>
      </FFF>
    </DDD>
  </XXX>
  <CCC>
    <DDD/>
  </CCC>
</AAA>
```

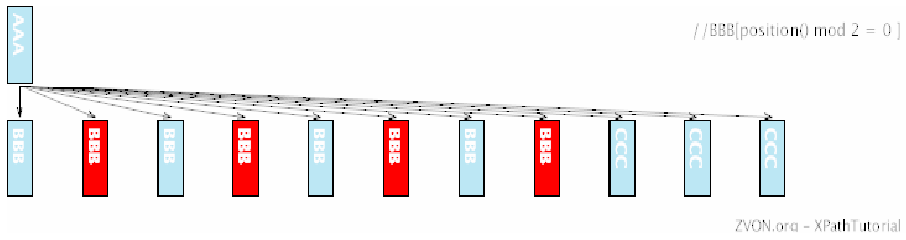

XPath - príklady www.zvon.org

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

vyber párne elementy BBB

matematická funkcia

```
<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <CCC/>
  <CCC/>
  <CCC/>
</AAA>
```

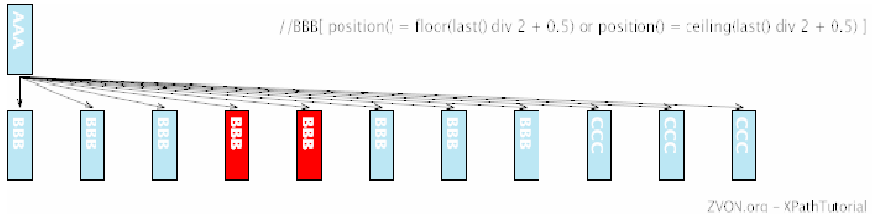


XPath - príklady www.zvon.org

//BBB[position() = floor(last() div 2 + 0.5) or position() = ceiling(last() div 2 + 0.5)]

vyber prostredné elementy BBB

```
<AAA>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <BBB/>
  <CCC/>
  <CCC/>
  <CCC/>
</AAA>
```



Preddefinované funkcie

základná množina funkcií, ktoré musí implementovať každý nástroj, pracujúci podľa špecifikácie XPath 1.0

spracovávanie existujúcich dátových typov:

- množina uzlov
- reťazce
- boolean
- čísla (reálne, plávavajúca desatinná čiarka)

Poznámka: neúplný list

Funkcie - množina uzlov

number **last()**
index posledného uzla v kontexte vyhodnotenia výrazu

number **position()**
index aktuálneho uzla v kontexte vyhodnotenia výrazu

number **count(***node-set***)**
počet uzlov v množine

Funkcie - množina uzlov

node-set **id(***object***)**
výber uzlov s určenými id-mi
nutné DTD
rôzne parametre (reťazcová reprezentácia)
príklad: id("foo")/para[position() = 3]

string **name(***node-set?***)**
meno (prvého) uzla, príp. kontextového uzla

Funkcie - reťazec znakov

string **string(***object?***)**
konvertuje objekt na reťazec

string **concat(***string,string,string**)**
vráti reťazcové spojenie argumentov

boolean **starts-with(***string,string***)**
vráti *true* ak prvý argument začína druhým argumentom

boolean **contains(***string,string***)**
vráti *true* ak prvý argument obsahuje druhý argument

Funkcie - reťazec znakov

string **substring(***string,number,number?***)**
vráti podreťazec od indexu (začína 1) a dĺžky

number **string-length(***string?***)**
vráti počet znakov v reťazci (default kontextový uzol)

string **normalize-space(***string?***)**
odstráni začiatočné a koncové whitespace znaky,
opakované konvertuje na znak medzery
(default kontextový uzol)

Funkcie - boolean

boolean **boolean(***object***)**
konvertuje objekt na boolean (neprázdne, nenulové, …)

boolean **not(***boolean***)**
negácia hodnoty parametra

boolean **true()** , *boolean* **false()**
vráti hodnotu *true* / *false*

Funkcie - čísla

number **number(***object***)**
konvertuje objekt na číslo

number **sum(***node-set***)**
súčet číselných hodnôt

number **floor(***number***)** , **ceiling** , **round**
úprava na celé číslo, zaokrúhľovanie

XPath - opakovanie

umožňuje:
- výber dát z XML dokumentu
- základnú manipuláciu s reťazcami, číslami, bool…

pracuje nad abstraktnou reprezentáciou XML dokumentu - **strom**

location path - osi, test uzla, predikáty

preddefinované funkcie
(množina uzlov, reťazec, boolean, číslo)

Ďakujem za pozornosť

Použitá (a študijná) literatúra

www.w3.org
oficiálne stránky WWW Consortia
(odporúčania sú na adresách namespace technológie)

www.w3schools.com
tutoriály k väčšine web-technológií

www.zvon.org
český server - veľa materiálov, tutoriály