Intro

XML och XSLT

Rasmus Kaj, Stacken <kaj@kth.se>
http://www.Stacken.kth.se/~kaj/

- Intro
- Korrekt XML?
- Alternativ till XSL?
- XSLT
- Funktioner
- Exempel / Problem

XML

XML är en syntax för markup, inte ett specifikt markupspråk.

- 1969: http://www.sgmlsource.com/history/
- 1981: SGML, SIGPLAN
- 1990: HTML, CERN / Tim Berners Lee
- 1996: XML draft
- 1998: XML 1.0

I dag används XML "överallt".

Korrekt XML?

Välformad XML?

Syntaxmässigt korrekt XML, utan hänsyn till vilka taggar som förekommer.

```
<?xml version="1.0" encoding="utf8"?>
<rootnode ...>
  <!-- kommentar ... -->
  <foo/>
  <bar type="important" color="red"> ... </bar>
  ... text ...
</rootnode>
```

Validerad XML

"Korrekta" taggar enligt någon instans:

- xhtml
- DocBook
- ...

Instansen specificeras i något formellt språk:

- DTD
- W3C Schema
- RelaxNG

Namnrymder

Flera namnrymder kan anges med prefix:

```
<foo xmlns="http://example.org/foo"
     xmlns:html="http://example.com/bar">
  <html:p class="example" xml:lang="sv">
    ... <bar/> ...
  </html:p>
</foo>
```

Entitys

Inbyggda

< & >

ቧ ቧ – Unicodenummer (oavsett dokumentets teckenuppsättning).

Övriga

ä &foobar; – Godtyckligt, måste definieras i angiven DTD. Används allt mindre.

Whitespace

Godtycklig mängd whitespace är ekvivalent med .

Whitespace "först och sist" kan tas bort

En omgivning kan vara whitespacebevarande

DTD

Document type definition

Måste anges explicit i dokumentet, antingen pekas ut eller inkluderas helt.

```
<?xml?>
<!DOCTYPE foo PUBLIC "-//KTH//Example//EN"
                     "http://example.org/example.dtd">
<foo> ... </foo>
```

RelaxNX

Pekas inte ut från dokumentet. Man anger schema när man validerar.

Man kan validera samma dokument med olika scheman (t ex vanlig html och en privat begränsad variant).

Alternativ till XSL?

CSS

- Varför XSL när det finns CSS?
  - CSS beskriver bara rendrering (utseende)
  - XSL kan göra godtyckliga omvandlingar.

Exempel: innehållsförteckning, index, RSS, ...

Gör "lagom" markup (t ex xhtml) med XSLT, speca utseende med CSS.

Specifika program

- Varför XSL när man kan skriva specifika program?

  • Det är mycket enklare!

... men ibland räcker det inte. Om man behöver koppla sin markupöversättning till andra program räcker det inte med XSLT (men man kan göra extensions).

Bibliotek: DOM, SAX

XSLT

XSL

Består av två delar:

XSLT

Ett sätt att översätta från ett tagspråk till ett annat.

XSL-FO

Ett taggspråk specifikt konstruerat för att beskriva hur innehållet ska se ut.

XSLT: Intro

En första enkel template

```
<xsl:template match="email">
  <a href="mailto:{.}">
    <xsl:apply-templates/>
  </a>
</xsl:template>
```

Stylesheet

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

  <xsl:include href="..."/>
  <xsl:param name="..."> ... </xsl:param>
  <xsl:variable name="..."> ... </xsl:variable>

  <xsl:template ...> ... </xsl:template>
  ...
</xsl:stylesheet>
```

Template

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

Anropas "automatiskt" när man träffar på en nod som matchar

```
<xsl:template name="bar">
   ...
<xsl:template>
```

Anropas explicit med:

```
<xsl:call-template name="bar">
```

Selectorer

match="...", select="..."

axel::bar – child, attribute, self, ancestor ...

foo/bar|foo//baz

foo[@bar = 'baz']

*[not(self::p)]

p[last()][@foo='bar']

p[@foo='bar'][last()]

Flödeskontroll

```
<apply-templates select="..."/>

<for-each select="..."> ... </for-each>

<if test="..."> ... </if>

<choose>
  <when test="..."> ... </when>
  <otherwise> ... </otherwise>
</choose>
```

Bygga resultat

"Direkt" resultat eller:

```
<element name="..."> ...</element>
<attribute name="..."> ... </attribute>
<text> ... </text>
<processing-instruction name="..."> ...
<comment> ... </comment>
<copy> ... </copy>
```

Funktioner

Strängfunktioner

- string(...)
- concat(x, y, z ...)
- substring(string, number, number?)
- contains(a, b)

Numeriska funktioner

- last()
- position()
- count()
- id()

Boolska funktioner

- not(...)
- true(), false()
- lang(...)

document(uri)

Retunerar det nodset som representeras av det dokument som pekas ut av parmetern

Om uri är relativ så utgår den från den aktuella xslfilen.

Exempel / Problem

Exempel: copy-all

```
<xsl:template match="@*|node()">
  <xsl:copy>
    <xsl:apply-templates select="@*|node()"/>
  </xsl:copy>
</xsl:template>
```

Vilken kontext?

```
<xsl:template match="abbr">
  <acronym title="{$db/str[@abbr=.]}">
    <xsl:apply-templates/>
  </acronym>
</xsl:template>
```

Både @abbr och . blir i str-context. Men vi vill ju ha . i abbr-context.

Vilken kontext?

Lösning:

```
<xsl:template match="abbr">
  <xsl:variable name="abbr" select="."/>
  <acronym title="{$db[@abbr=$abbr]}">
    <xsl:apply-templates/>
  </acronym>
</xsl:template>
```

Referens

Lite mer att läsa:

- http://www.w3.org/
- http://www.w3.org/TR/xslt
- http://www.w3.org/TR/xpath
- http://www.sgmlsource.com/history/
- http://exslt.org/
- http://www.stacken.kth.se/~kaj/