Rethinking XSLT

dntw3c.org

Sam Wilmott

Extreme Markup 2006

Rethinking XSLT

Notation Is Important

White Space Is Important

Sam Wilmott
Extreme Markup 2006

Notation Is Important

- For common understanding:
  - of the data being processed and transmitted, and
  - of the processes being applied to the data.
- For productively efficient use:
  - of that data, and
  - of those processes.

Notation Is Important

In the context of XSLT, there are four notations to consider:

- The notation of the XML input data.
- The notation of the XML or other output data.
- The notation of the instruction logic of the XSLT stylesheet.
- The notation of the template data (result elements) within an XSLT stylesheet.

Notation Is Important

- Depending on the way you look at things:
  - XML is a notation.
  - XML is a meta-notation.
- Either way, XML is about notation, and like any other notation:
  - XML is a good (meta-)notation for some (a monstrous big lump of some) things.
  - XML is not so good for some (a monstrous also, if not so much so as XML, lump of some) things.

Coding Progam Logic in XML

dntw3c.org

Sam Wilmott

Extreme Markup 2006

XML Encoding C

```
<cml:program version=”current c version”
    xmlns:cml=”uri for c language standard”
    xmlns:up=”uri for this user’s programs”>
  <cml:function name=”f” as=”cml:int”>
    <cml:param name=”n” as=”cml:int”/>
    <cml:if test=”$n=0”>
      <cml:then>
        <cml:return select=”0”/>
      </cml:then>
      <cml:else>
        <cml:return select=”$n+1”/>
      </cml:else>
    </cml:if>
</cml:function>
</cml:program>
```

XML Encoding C

Here's the same thing in the usual C notation:

```
int f (int n)
{
  if (n = 0)
    return 1;
  else
    return n + 1;
}
```

XML Encoding C

Where the traditional C differs primarily from the XML-encoded C version is in the following:

- There's no self-identification or versioning of the notation/language being used (no version= or xmlns).
- Every language component is not explicitly identified as being part of the language (no xsl: or cml:).
- There are a minimum of notational artifacts in the language: if's argument is a test, it doesn't need saying.

XML Encoding C

- Advantages of the C approach:
  - Easier to read.
  - Less typing.
  - Minimally redundant.
- Advantages of the XSLT approach:
  - Maximizes the information available to an XSLT processor.
  - Supports XML template data (result elements).

And Now For Something (Not) Completely Different

Learning from C

Where could XSLT learn a lesson from C?

- NOT in copying C.
- In looking at an XSLT stylesheet as what it is: a program.
- In removing all the XML artifacts from the non-XML parts of an XSLT program.
- In approximating a notation that people naturally use. (Which is what C did, 30+ years ago, with different people.)

Seen on White Boards

```
template “chapter/title”
  element = “H1”


xsl:attribute “ALIGN”>CENTER</>


xsl:if position()=1
   {  attribute “indent”       0  }
```

RXSLT

Removing the XML-encoding artifacts:

```
variable first-chapter-tag = “(Chapter) “

template chapter/title
  element H1
    attribute ALIGN
      “CENTER”
  if ../position () = 1
    value-of $first-chapter-tag
  apply-templates
```

RXSLT

Result elements fit right in:

```
template para
  <P>
    {apply-templates}
  </P>

template chapter/title
  <H1 ALIGN=”CENTER>
    {apply-templates}
  </H1>
```

RXSLT

XPath expressions become part of the language: no quoting needed.

```
template example
  if not (parent-or-ancestor::annex or
          parent-or-ancestor::front-matter)
    <PRE>{value-of text-example}</PRE>
```

White Space Is Important

White space is the dirty secret of markup languages.

- SGML did it one way.
- XML does it two ways (preserve/default), neither the same as SGML's way.
- Specific markup applications have their own appropriate rules.
- All are good sometimes. All are bad sometimes. One size does not fit all.

White Space Is Important

White space is important in different contexts:

- For making a program/stylesheet readable.
- For making input data readable.
- In the presentation form of output data (for print/web etc. applications).

Each context has its own requirements. Mixing the requirements results in conflict, difficulty, and grief.

White Space Is Important

Seen in some familiar XSLT stylesheets:

```
<xsl:template select="email">
  <bold>
    <xsl:text>[</xsl:text>
    <xsl:apply-templates/>
    <xsl:text>]</xsl:text>
  </bold>
</xsl:template>

<xsl:template select=”name”>
  <xsl:value-of select=”first”/>
  <xsl:text> </xsl:text>
  <xsl:value-of select=”last”/>
</xsl:template>
```

White Space Is Important

A simple RXSLT equivalent:

```
template email
  <bold>
    {
      "["
      apply-templates
      "]"
    }
  </bold>
```

White Space Is Important

And the other one:

```
template name
  value-of first
  " "
  value-of last
```

Sam Wilmott
Extreme Markup 2006

White Space Is Important

Some better RXSLT equivalents:

```
template email
  <bold>[{
      apply-templates
  }]</bold>

template email
  <bold>[{apply-templates}]</bold>

template name
  value-of first; “ “; value-of last
```

White Space Is Important

Moral:

In a template programming language:

You need syntactic separation of the program logic and the template data.

Back to RXSLT

- Implements XSLT 1.0. (Got to start somewhere.)
- There's a fully working implementation written in Python. (Just because it's good for getting things up and running fast.)
- It translates RXSLT into XSLT.
- Took about a week to implement the whole of XSLT 1.0, minus getting the white space in the output right.
- White space took a couple of weeks.
- Examples took a few more days.

RXSLT

Another moral:

In spite of all the XSLT composition and editing tools available, a lot of XSLT stylesheets are written by people.

We need to think more of the people.

RXSLT?

“Rethought XSLT”

“Revised XSLT”

“Real XSLT”

dntw3c.org

Definitely Not The w3c.org

Where Is It?

www.wilmott.ca/rxslt Now

www.dntw3c.org Soon