Jesper Tverskov, April 12, 2008

Understanding Processing Instructions in XML

Processing Instructions are special tags with instructions to software making us of the XML document. It sounds important and makes PIs extremely confusing for novices. The truth is that PIs are seldom used with a few common exceptions

1. Why Processing Instructions?

It HTML we have no Processing Instructions. When the need to send instructions to software arose, developers were forced to misuse the comment tag.

Server Side Includes (SSI) is the classical example of misuse of the HTML comment tag for a good purpose. A typical SSI to include a file could look like this: [1]

```
<!--#include virtual="somefile.txt" -->
```

It might be smart but it is misuse of the comment tag. If the HTML page used "shtml" as file extension, it was a signal to the web server to look for SSI and process them before the page was sent to the browser.

Microsoft also invented "conditional comments" to make it easy to serve optimized markup and CSS for IE browsers: [2]

```
<!--[if IE 7]>
<link rel="stylesheet" type="text/css" href="ie7only.css">
<![endif]-->
```

It might be smart but it is misuse of the comment tag for something that is not a comment.

So that was how the working group making the XML spec got the idea that developers can have special needs impossible to foresee. Compared to HTML being a markup language, special needs are much more likely in XML being a meta markup language to make markup languages.

The PI tag was invented to protect the comment tag from misuse and in order to provide a flexible universal mechanism for all sorts of special needs not easily solved with elements and attributes alone.

2. Processing Instruction

Processing Instruction is defined in the XML standard, XML 1.0 Recommendation. The definition says: "Processing instructions (PIs) allow documents to contain instructions for applications… PIs are not part of the document's character data, but MUST be passed through to the application. "

An XML processor should simply ignore the meaning of all Processing Instructions because the processor has no way of knowing what they mean. But the XML processor must pass the PI's name and content on to the application.

The XML spec also says:

```
[16] PI ::= '<?' PITarget (S (Char* - (Char* '?>' Char*)))? '?>'
[17] PITarget ::= Name - (('X' | 'x') ('M' | 'm') ('L' | 'l'))
```

The name or the so-called PI Target following the first question mark is the name of the PI. It must be followed by a space, and the rest of the PI until the last question mark, is the content that is just some string value. The letters "x", "m" and "l" are reserved and must not start a user-defined PI name. The content must not contain the letters "?>".

Notice that the content of a PI is just some string value. But it is very common to make content that looks like pseudo-attributes. An example is the content of the xml-stylesheet PI (see later):

```
<?xml-stylesheet href="mystyle.css" type="text/css"?>
```

But the following could also be legal content: "PIs are a little strange until you get used to them" or "xyz7410293".

3. Known uses of PIs

In the real world PIs are seldom used with a few common exceptions. The PI is the official method for an XML document to link to a stylesheet, and Microsoft uses PIs in MS Office 2003 to make it easy for MS Word and MS Excel documents to be identified when saved as XML.

3.1 xml-stylesheet

The use of Processing Instructions for linking of stylesheets to XML document became a standard, a W3C Recommendation, in 1999, Associating Style Sheets with XML documents:

```
<?xml-stylesheet href="mystyle.xslt" type="text/xsl"?>
```

3.2 PI in MS Office 2003

With MS Office 2003 the use of PIs suddenly exploded. Both Excel and Word can save in their own xml formats using the "xml" file extension. This was a terrible mess because many other XML applications also use the "xml" extension. [3]

In order to signal to Internet Explorer and to Windows Explorer that an XML file was a Word or an Excel document, Microsoft used PIs for identification.

For MS Word 2003 the PI situated right after the XML declaration looks like this:

```
<?mso-application progid="Word.Document"?>
```

For MS Excel 2003 the PI looks like this:

```
<?mso-application progid="Excel.Sheet"?>
```

MS Office 2007 uses zipped file format for Word (docx), Excel (xlsx) and PowerPoint (pptx). Each zipped document contains several XML files, media files, etc. In MS Office 2007 there is no need to use PIs, the new file extensions are enough. PIs are still used if you in Office 2007 save in the 2003 XML formats.

4. PIs for anything

In text books and in tutorials we can find all sorts of often far out usages of PIs: They can be used to indicate line breaks in some sorts of markup, for indicating what should not be spell checked, etc.

Let me give you another example as a reminder of the diversity of problems PIs could solve. Let us say we have some XML project. The project team could decide to use Processing Instructions as a way to annotate XML documents still in the making:

```
<?TeamAlpha member="Jesper" date="2008-04-15" comment="I strongly
feel that the attributes in the product element below are really not
needed and should be dropped." ?>
```

If comment tags were used they could be difficult to distinguish from real comments of the document. PIs could come in handy: We don’t need to declare them in the schema, we can easily tell them apart from traditional comments, and we can easily delete them again when time comes.

A comment as defined in the XML spec only has content. PIs have the advantage of also having a name. An example: In XPath we have the processing-instuction() node test matching all PIs. But we can also find PIs by name and get their content returned:

```
processing-instruction()[name()="TeamAlpha"]
```

Remember that the content is just one long string. The pseudo values of the pseudo attributes can be analyzed with Regular Expressions.

5. When to use PIs

Regard PIs as a last resort. Don’t use them if they are not needed. PIs will be ignored if software is not told what they mean. Unless PIs are easy to understand like the xml-stylesheet PI and the PIs in the MS Office 2003 XML file formats, even developers must be told what they mean before they make sense.

6. XML declaration is not a PI

Many text books and tutorials will tell you that the XML declaration is a Processing Instruction or a special PI. It drives me mad, and I usually blacklist such books. It is hard to believe anything in a book about XML if the author can not even tell XML declaration and PI apart.

An XML declaration, <?xml version="1.0" encoding="utf-8" ?>, looks like a PI, and it very much acts like a PI except that the XML processor knows the XML declaration and what it can contain by heart and returns an error message if anything in it is wrong.

The whole idea about PIs as stated in the XML spec is the exact opposite: that the XML processor should ignore PIs except that their name and content must be passed on to the application.

In XSLT we can create PIs with the xsl:processing-instruction element but we can not use it to create an XML declaration. It is not a PI. In XPath we have a processing-instruction() node tests but is can not find and return the content of the XML declaration not being a PI.

Footnotes

[1]

Wikipedia: Server Side Includes.

[2]

Wikipedia: Conditional comment.

[3]

I have used the MS Word 2003 XML file format a lot, and to give myself a chance to remember what XML documents belonged to Word, I used the traditional "doc" in the filename like this: mydocument.doc.xml.

Updated: 2011-08-02