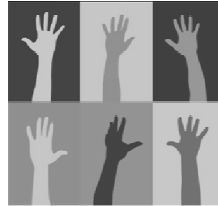
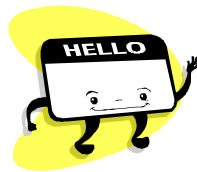


Looking Inside the Developer's Toolkit: Introduction to XML and RPG



Charles Guarino

Central Park Data Systems, Inc.

1

Copyright Central Park Data Systems, Inc.

About The Presenter

With an IT career spanning over 30 years, Charles Guarino has been a consultant for most of them. Since 1995 he has been founder and President of Central Park Data Systems, Inc., a New York area based IBM midrange consulting company. He has published several technical and strategic articles for the IT community and continues in that role as a frequent contributor of programming and developer tips to System i Network and Power IT Pro. He is a proud member of COMMON's Speaker Excellence Hall of Fame and also Long Island Software and Technology Network's Twenty Top Techies of 2009. Charles currently serves as a COMMON Application Development Subject Matter Expert and President and monthly Q&A host of LISUG, a Long Island IBM System i User's Group www.lisug.org. Charles can be reached at cguarino@centralparkdata.com. LinkedIn - <http://www.linkedin.com/in/guarinocharles>

What We'll Cover

- **Essential Definitions**
- **XML – A First Look**
- **A Review of Parsers**
- **XML-INTO: A Closer Look**
- **XML-SAX: A Closer Look**
- **Wrap-up**

Copyright Central Park Data Systems, Inc.

3

ESSENTIAL DEFINITIONS

- 1) What is XML and why use it?
- 2) Element
- 3) Attribute
- 4) Parsers – 2 types!
- 5) Well formed document
- 6) Where is XML typically stored on the IBM i?

Copyright Central Park Data Systems, Inc.

4

What We'll Cover

- **Essential Definitions**
- **XML – A First Look**
- **A Review of Parsers**
- **XML-INTO: A Closer Look**
- **XML-SAX: A Closer Look**
- **Wrap-up**

Copyright Central Park Data Systems, Inc.

5

If you are already familiar with HTML...

The screenshot shows the w3schools.com website. At the top, there's a logo and a search bar. Below the logo, the text 'w3schools.com' is displayed. To the right of the logo, there's a 'TRANSLATE' button and a search bar. The main content area is titled 'HTML Tutorial' and includes a welcome message: 'With HTML you can create your own Web site. This tutorial teaches you everything about HTML. HTML is easy to learn - You will enjoy it.' Below this, there's a section titled 'Examples in Each Chapter' with a text area containing HTML code and a 'Try it yourself' button. The right sidebar contains several sections: 'WEB HOSTING' with links to various hosting services, 'WEB BUILDING' with links to various building tools, 'W3SCHOOLS EXAMS' with a link to get certified, 'W3SCHOOLS BOOKS' with a link to new books, 'STATISTICS' with a link to browser statistics, and 'SHARE THIS PAGE' with a link to share the page.

Copyright Central Park Data Systems, Inc.

Then XML will be a snap!

w3schools.com TRANSLATE
Search w3schools.com Search

HOME HTML CSS XML JAVASCRIPT ASP PHP SQL MORE... REFERENCES | EXAMPLES | FORUM | ABOUT

XML Basic

- XML HOME
- XML Introduction
- XML How to use
- XML Tree
- XML Syntax
- XML Elements
- XML Attributes
- XML Validation
- XML Validator
- XML Viewing
- XML CSS
- XML XSLT

XML JavaScript

- XML HTTP Request
- XML Parser
- XML DOM
- XML to HTML
- XML Applications

XML Advanced

- XML Namespaces
- XML CDATA
- XML Encoding
- XML Server
- XML DOM Advanced
- XML Don't
- XML Technologies
- XML in Real Life
- XML Editors
- XML Summary

XML Examples

- XML Examples
- XML Quiz
- XML Certificate

XML Tutorial

« W3Schools Home Next Chapter »

XML stands for **eXtensible Markup Language**.

XML is designed to transport and store data.

XML is important to know, and very easy to learn.

Start learning XML now!

XML Document Example

```
<?xml version="1.0"?>
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

XML Examples

Learn by example! With our editor, you can edit XML and click on a test button to view the result.

[Try-it-yourself!](#)

[XML Quiz Test](#)

WEB HOSTING

- Best Web Hosting
- PHP MySQL Hosting
- Top 10 Web Hosting
- UK Reseller Hosting
- Cloud Hosting
- \$5.93 Domain w/ Extras
- Cheap Web Hosting

WEB BUILDING

- XML Editor - Free Trial!
- Free HTML Website
- Free Website Templates
- Free WordPress Themes

W3SCHOOLS EXAMS

- Get Certified in: HTML, CSS, JavaScript, XML, PHP, and ASP

W3SCHOOLS BOOKS

- New Books: HTML, CSS, JavaScript, and Ajax

STATISTICS

- Browser Statistics
- Browser OS
- Browser Display

SHARE THIS PAGE

- Share with »

Copyright Central Park Data Systems, Inc.

Document "citydata1.xml"

IFS Files

- File systems
- Root file system
- Home
- xml docs
 - citydata.xml
 - citydata1.xml**
 - citydata2.xml

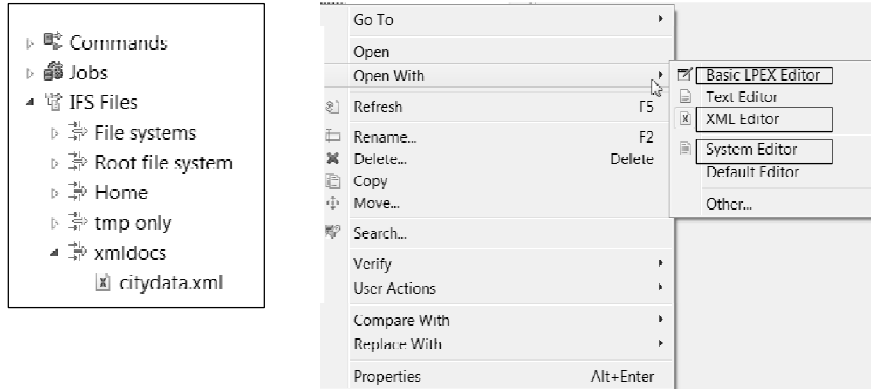
citydata1.xml

Line	Column	Insert
1	1	2

```
<Cities>
<CityData>
  <CityName>New York</CityName>
  <Region>Northeast</Region>
  <MonthlyData>
    <Month>May</Month>
    <Low>53</Low>
    <High>71</High>
  </MonthlyData>
  <Month>June</Month>
  <Low>63</Low>
  <High>81</High>
  </MonthlyData>
  <MonthlyData>
    <Month>July</Month>
    <Low>68</Low>
    <High>81</High>
  </MonthlyData>
</CityData>
<CityData>
  <CityName>Chicago</CityName>
  <Region>Midwest</Region>
  <MonthlyData>
    <Month>May</Month>
    <Low>51</Low>
    <High>70</High>
  </MonthlyData>
  <MonthlyData>
    <Month>June</Month>
    <Low>61</Low>
    <High>80</High>
  </MonthlyData>
  <MonthlyData>
    <Month>July</Month>
    <Low>66</Low>
    <High>84</High>
  </MonthlyData>
</CityData>
</Cities>
```

Copyright Central Park Data Systems, Inc.

How do you want to view the XML data using RDP?



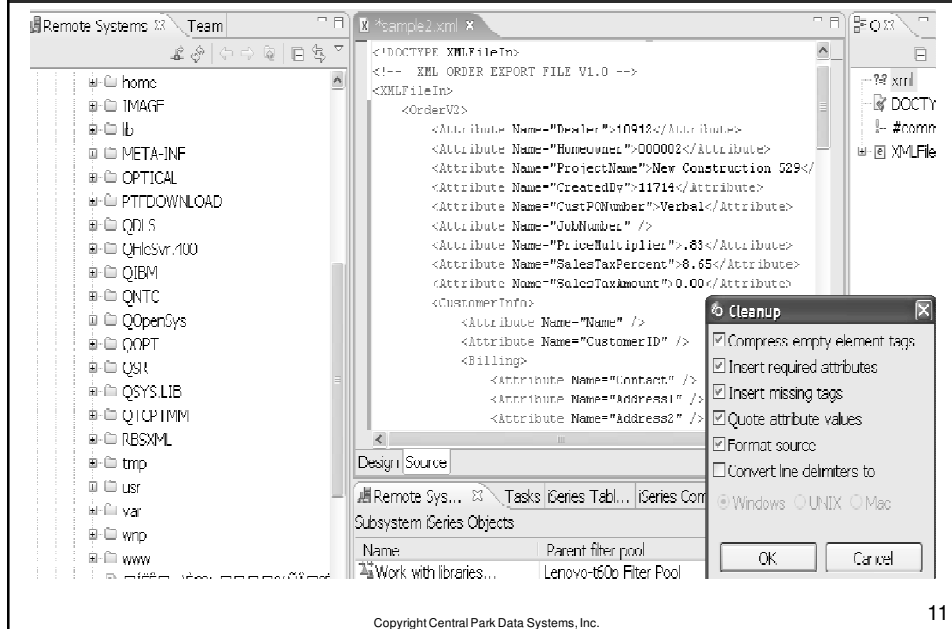
Copyright Central Park Data Systems, Inc.

XML beauty is in the eyes of the beholder...

citydatashort.xml	citydatashort.xml	C:\cgonly\CPD's workspace\Remotesy...
<pre> Line 1 Column 1 insc + + + + + + + + <Cities> <CityData> <CityName>New York</CityName> <Region>Northeast</Region> <MonthlyData> <Month>May</Month> <Low>53</Low> <High>71</High> </MonthlyData> <MonthlyData> <Month>June</Month> <Low>63</Low> <High>81</High> </MonthlyData> <MonthlyData> <Month>July</Month> <Low>68</Low> <High>81</High> </MonthlyData> </CityData> </Cities> </pre>	<pre> Node +-----+ Cities +-----+ CityData +-----+ CityName +-----+ Region +-----+ MonthlyData +-----+ MonthlyData +-----+ MonthlyData +-----+ </pre>	<pre> - <Cities> - <CityData> <CityName>New York</CityName> <Region>Northeast</Region> - <MonthlyData> <Month>May</Month> <Low>53</Low> <High>71</High> </MonthlyData> - <MonthlyData> <Month>June</Month> <Low>63</Low> <High>81</High> </MonthlyData> - <MonthlyData> <Month>July</Month> <Low>68</Low> <High>81</High> </MonthlyData> </CityData> </Cities> </pre>
Basic LPEX Editor	XML Editor	System Editor (browser)

Copyright Central Park Data Systems, Inc.

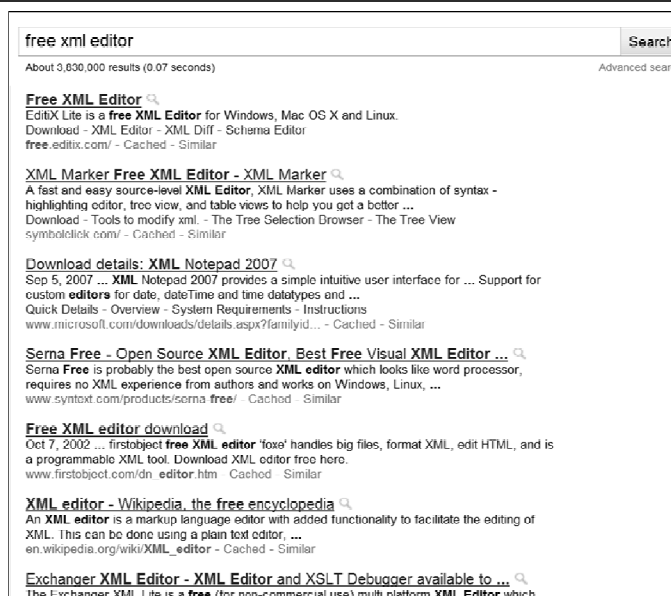
One benefit of using an XML Editor – The ‘Cleanup’ feature!



Copyright Central Park Data Systems, Inc.

11

If you don't like the choices directly available in RDP...



Copyright Central Park Data Systems, Inc.

What We'll Cover

- **Essential Definitions**
- **XML – A First Look**
- **A Review of Parsers**
- **XML-INTO: A Closer Look**
- **XML-SAX: A Closer Look**
- **Wrap-up**

Copyright Central Park Data Systems, Inc.

13

Which parser to use?

Examine the XML document.

Does it have a consistent structure, with repetitive groups or many different structures, and/or complex structures?

Will you know in advance how the XML is actually formatted?

How will you be using the data?
Is it a particularly large document?

These important answers will direct you to which parser to use.

Copyright Central Park Data Systems, Inc.

14

DOM = Document Object Model

Allows an application to read and update XML data directly in memory. In the DOM implementation, data is moved into arrays and data structures.

Programmers need to be sensitive to how large an XML document is so as to not exceed the available system storage. Optionally, a 'handler' can be used to deal with array overflow.

RPG implementation  "XML-INTO"

Copyright Central Park Data Systems, Inc.

15

SAX = Simple API for XML

The SAX parser walks (runs?) through an entire XML document, small section by small section at a time, and returns control to the handler as each new 'event' is encountered. The complexity of the document is not relevant.

Also, since system memory is not consumed, the document size is not relevant.

RPG implementation  "XML-SAX"

Copyright Central Park Data Systems, Inc.

16

Where can you find sample XML? There are lots of documents to be downloaded. Take this fairly simple one.

```

http://www.w3schools.com/xml/simpl...

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- Edited by XMLSpy@ -->
- <breakfast_menu>
- <food>
  <name>Belgian Waffles</name>
  <price>$5.95</price>
  <description>two of our famous Belgian Waffles with plenty of real maple syrup</description>
  <calories>650</calories>
</food>
- <food>
  <name>Strawberry Belgian Waffles</name>
  <price>$7.95</price>
  <description>light Belgian waffles covered with strawberries and whipped cream</description>
  <calories>900</calories>
</food>
- <food>
  <name>Berry-Berry Belgian Waffles</name>
  <price>$8.95</price>
  <description>light Belgian waffles covered with an assortment of fresh berries and whipped cream</description>
  <calories>900</calories>
</food>
- <food>
  <name>French Toast</name>
  <price>$4.50</price>
  <description>thick slices made from our homemade sourdough bread</description>
  <calories>600</calories>
</food>
- <food>
  <name>Homestyle Breakfast</name>
  <price>$6.95</price>
  <description>two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
  <calories>950</calories>
</food>
</breakfast_menu>

```

Copyright Central Park Data Systems, Inc.

This complex XML document was computer-generated, 75 pages long.

```

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE XMLFileIn [View Source for full doctype...]>
<!-- XML ORDER EXPORT FILE V1.0 -->
- <XMLFileIn>
- <OrderV2>
  <Attribute Name="Dealer">10912</Attribute>
  <Attribute Name="Homeowner">0000002</Attribute>
  <Attribute Name="ProjectName">New Construction 529</Attribute>
  <Attribute Name="CreatedBy">11714</Attribute>
  <Attribute Name="CustPONumber">Verbal</Attribute>
  <Attribute Name="InhNumber" />
  <Attribute Name="PriceMultiplier">.83</Attribute>
  <Attribute Name="SalesTaxPercent">0.65</Attribute>
  <Attribute Name="SalesTaxAmount">0.00</Attribute>
  <CustomerInfo>
    <Attribute Name="Name" />
    <Attribute Name="CustomerID" />
  </CustomerInfo>
  <Billing>
    <Attribute Name="Contact" />
    <Attribute Name="Address1" />
    <Attribute Name="Address2" />
    <Attribute Name="City" />
    <Attribute Name="State" />
    <Attribute Name="ZipCode" />
    <Attribute Name="Phone" />
  </Billing>
  <Shipping>
    <Attribute Name="Contact" />
    <Attribute Name="Address1" />
    <Attribute Name="Address2" />
  </Shipping>
</OrderV2>
</XMLFileIn>

```

This document contains over 150 different complex structures. SAX is clearly the way to go!

Copyright Central Park Data Systems, Inc.

18

What We'll Cover

- Essential Definitions
- XML – A First Look
- A Review of Parsers
- XML-INTO: A Closer Look
- XML-SAX: A Closer Look
- Wrap-up

Copyright Central Park Data Systems, Inc.

19

Introducing...

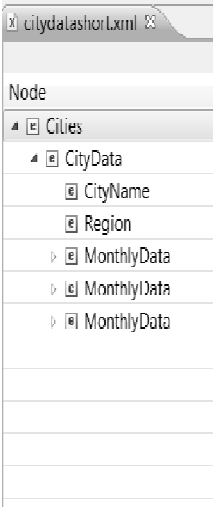


XML-INTO !!!

- You need to know the structure format in advance
- Data is mapped into data structures
- Can parse an entire document at one time
- Can use an optional handler for every large documents
- Recommended for less complex structured documents

Copyright Central Park Data Systems, Inc.

When using XML-INTO, the XML editor can help you define your data structures



The XML editor shows a tree view on the left with the following structure:

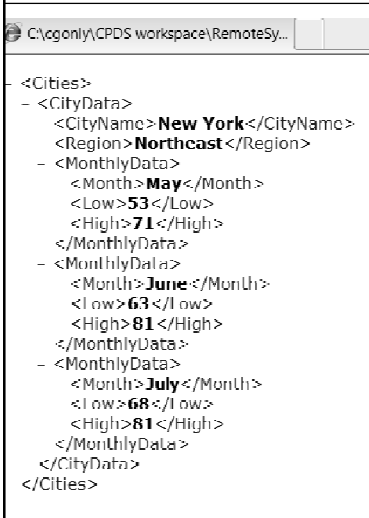
- Node
 - Cities
 - CityData
 - CityName
 - Region
 - MonthlyData
 - MonthlyData
 - MonthlyData
 - MonthlyData

The data structure table on the right is as follows:

D Cities	ds	qualified
D CityData		likeds(citydata) dim(3)
D		
D CityData	ds	qualified
D CityName		20
D Region		20
D MonthlyData		likeds(MonthlyData) dim(3)
D MonthlyData	ds	
D Month		9
D Low		3
D High		3

Copyright Central Park Data Systems, Inc.

It's also helpful to use the "tag", "data", "tag" approach



The XML editor shows an XML document on the left with the following structure:

```

<Cities>
  <CityData>
    <CityName>New York</CityName>
    <Region>Northeast</Region>
    <MonthlyData>
      <Month>May</Month>
      <Low>53</Low>
      <High>71</High>
    </MonthlyData>
    <MonthlyData>
      <Month>June</Month>
      <Low>63</Low>
      <High>81</High>
    </MonthlyData>
    <MonthlyData>
      <Month>July</Month>
      <Low>68</Low>
      <High>81</High>
    </MonthlyData>
  </CityData>
</Cities>
  
```

The data structure table on the right is as follows:

D Cities	ds	qualified
D CityData		likeds(citydata) dim(3)
D		
D CityData	ds	qualified
D CityName		20
D Region		20
D MonthlyData		likeds(MonthlyData) dim(3)
D MonthlyData	ds	
D Month		9
D Low		3
D High		3

Copyright Central Park Data Systems, Inc.

Our first XML parsing program “citydata1”

```

CITYDATA1.RPGLE
Line 1      Column 1      Replace
.....1.....2.....3.....4.....5.....6.....7.....8.....9.....
000200      Fcitydata1 0 a u          disk
000300
000400
000500      D Cities          ds          qualified
000600      D CityData          ds          liked(citydatads) dim(2)
000700      D
000800      D CityDatads        ds          qualified
000900      D CityName          ds          20
001000      D Region          ds          20
001100      D State          ds          20
001200      D MonthlyData    ds          liked(MonthlyData) dim(3)
001300
001400      D MonthlyData    ds
001500      D Month          ds          9
001600      D Low          ds          3
001700      D High          ds          3
001800
001900      D optional1        s          100a
002000      D filename        s          25a inz('/xmldocs/citydata1.xml') varying
002001      D x          s          2 0
002002      D y          s          2 0
002100
002200      /free
002300
002400      // Setup parsing options
002500      optional = 'doc=file +
002600      allowextra=yes allowmissing=yes case=any';
002700
002800      // Initialize path to file in IFS
002900      filename = '/xmldocs/citydata1.xml';
003000
003100      // Capture header data into data structure Cities
003200      xml-into Cities &xml(filename:optional1);
003300
003400      // Capture detail data, load subfile
003500      for x = 1 to 2;
003600      cityname = cities.citydata(x).cityname;
003700

```

Copyright Central Park Data Systems, Inc.

“Show in Table” view of parsed data as it resides in physical file CITYDATA1

CPDS XML ConnectionXMLLIB/CITYDATA1(CITYDATA1)					
*RC1JNRR	CITYNAME	REGION	MONTHNAME	LOW	HIGH
1	New York	Northeast	May	53	71
2	New York	Northeast	June	63	81
3	New York	Northeast	July	68	81
4	Chicago	Midwest	May	51	70
5	Chicago	Midwest	June	61	80
6	Chicago	Midwest	July	66	84

Copyright Central Park Data Systems, Inc.

What is a handler?

```
exsr mathsubroutine;  
x = y + z;  
{Mainline code here}  
  
begsr mathsubroutine;  
.  
endsr;
```

```
xml-into %handler(XMLHandler:MyCommArea) %xml(filename:options1);
```

```
x = y + z;  
{Mainline code here}
```

```
P XMLHandler    b
```

```
{Handler code here}
```

```
P XMLHandler    e
```

Copyright Central Park Data Systems, Inc.

Using a handler to parse very large documents

```
D Cities          ds          qualified  
D CityData        ds          likeds(CityData) dim(3)  
D  
D CityData        ds          qualified  
D CityName        20  
D Region          20  
D MonthlyData     ds          likeds(MonthlyDataDS) dim(3)  
  
D MonthlyDataDS   ds          9  
D Month           3  
D Low             3  
D High            3
```

CityData is defined as dim(3).
This structure repeats 5 times.
Cities are New York,
Chicago, Dallas,
Miami and San Francisco.
This could easily have been
10,000 worldwide cities.
This will require 2 passes of
parsing.



```
CityData2.xml 83  
Line 33      Column 109      Insert  
-----1-----2-----3-----4-----  
<Cities>  
<CityData>  
<CityName>New York</CityName>  
<Region>Northeast</Region>  
<MonthlyData>  
<Month>May</Month>  
<Low>53</Low>  
<High>71</High>  
</MonthlyData>  
<MonthlyData>  
<Month>June</Month>  
<Low>63</Low>  
<High>81</High>  
</MonthlyData>  
<MonthlyData>  
<Month>July</Month>  
<Low>68</Low>  
<High>81</High>  
</MonthlyData>  
</CityData>
```

Copyright Central Park Data Systems, Inc.

Enter “datasubf”, available in 7.1 or as a 6.1 PTF

```

STATSRPGLE
Line 1      Column 1      Replace
-----+-----+-----+-----+-----+-----+-----+-----
000700
002805      D states      ds      qualified
002806      D state      ds      likeda(stateds) dim(99)
002807
002809      D stateds      ds
002013      D abbreviation      2
002814      D capital      20
002815      D statename      30
003400
005400
005500      D options1      S      100a
005601      D filename      S      25a      inz('xmldocs/states.xml')varying
005700
005800
005900      /frc
006000
006001      options1 = 'doc-file 1
006002      allowextra=yes allowmissing=yes case=any datasubf=statename';
009700
009800      // Capture header data
009900      xml-into states %xml(filename:options1);
010000
010001      *inlr = *on;
012700
013900

```

Copyright Central Park Data Systems, Inc

XML-INTO can now parse complex XML tag structures !

RPG Café

Proven Productivity

[Cafés](#)
[Resource Library](#)
[Discussion Forums](#)
[Blogs](#)
[Hubs](#)
[Products](#)

[IBM Rational Café](#) > [RPG Café](#) > Documents

[Up to Documents in RPG Café](#)

Two new options for XML-INTO with a V6R1 PTF

Created on: Mar 5, 2009 2:18 PM by barbara_morris - Last Modified: Mar 27, 2009 3:48 PM by barbara_morris

Two new options for XML-INTO

V6R1 PTF SI34938 adds two new %XML options for the XML-INTO operation. To take advantage of this enhancement, RPG programs will have to be recompiled with PTF SI34941.

The new options are

- `datasubf`
- `countprefix`

The `datasubf` option enables the XML-INTO operation to handle XML elements in the following form:

```
<tagname attr="value">text data</tagname>
```

The `countprefix` option provides a way to detect the number of subfields and subfield array elements which were set by the XML-INTO operation. It reduces the need to specify the `allowmissing=yes` option.

Copyright Central Park Data Systems, Inc

30

Namespaces handling: New for 7.1 or as a PTF in 6.1

You are in: RPG Cafe > Welcome > XML-INTO enhancement for namespaces and other non-RPG names

XML-INTO enhancement for namespaces and other non-RPG names

☆ 0

Updated Feb 01 by barbara_morris | Tags: None

Page Actions ▾

6.1 PTF SI42426 provides support for some new XML-INTO options to deal with namespaces and with other XML names that do not map to RPG names.

Update 2011-02-01 The 7.1 support is provided by PTF is SI42510.

Tiny URL for this page: <http://preview.tinyurl.com/rpg-xml-into-namespace>



- There are two new XML-INTO options to handle XML names containing a namespace.
 - The namespace option, *ns*, controls how XML-INTO handles XML names with a namespace when it is matching XML names to the names in the path option or the subfield names of a data structure.
 - The namespace prefix option, *nsprefix*, allows your RPG program to find out the values of the namespaces that were removed from the XML names when the *ns=remove* option was used to remove the namespace from the names.
- Also, there is a new value for the case option, *case=convert*, to handle non-RPG characters in XML names

~~A 7.1 PTF will be available soon. This page will be updated with information about the 7.1 PTF as soon as it is available.~~
The 7.1 PTF is SI42510.

See the attachment [xml-into-namespace-options.pdf](#) for more details about these new options.

Sample XML with namespaces

```
<root
xmlns:h="http://www.w3.org/TR/html4/"
xmlns:f="http://www.w3schools.com/furniture">

<h:table>
  <h:tr>
    <h:td>Apple</h:td>
    <h:td>Bananas</h:td>
  </h:tr>
</h:table>

<f:table>
  <l:name>African Coffee Table</l:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>

</root>
```

Source: http://www.w3schools.com/xml/xml_namespaces.asp

Error codes for RNX0351 (also RNX0352 and RNX0353)

XML Parser Error Codes

If the XML parser detects an error in the XML document during parsing, message RNX0351 will be issued. From the message, you can get the specific error code associated with the error, as well as the offset in the document where the error was discovered.

The following table shows the meaning of each parser error code:

XML Parser Error Code	Description
1	The parser found an invalid character while scanning white space outside element content.
2	The parser found an invalid start of a processing instruction, element, comment, or document type declaration outside element content.
3	The parser found a duplicate attribute name.
4	The parser found the markup character '<' in an attribute value.
5	The start and end tag names of an element did not match.
6	The parser found an invalid character in element content.
7	The parser found an invalid start of an element, comment, processing instruction, or CDATA section in element content.
8	The parser found in element content the CDATA closing character sequence ']]>' without the matching opening character sequence '<![CDATA['.
9	The parser found an invalid character in a comment.
10	The parser found in a comment the character sequence '--' (two hyphens) not followed by '>'.
11	The parser found an invalid character in a processing instruction data segment.
12	A processing instruction target name was 'xml' in lowercase, uppercase or mixed case.
13	The parser found an invalid digit in a hexadecimal character reference (of the form ෝ for example ັ).
14	The parser found an invalid digit in a decimal character reference (of the form &#ddd;).
15	A character reference did not refer to a legal XML character.
16	The parser found an invalid character in an entity reference name.
17	The parser found an invalid character in an attribute value.
18	The parser found a possible invalid start of a document type declaration.
19	The parser found a second document type declaration.
20	An element name was not specified correctly. The first character was not a letter, '_', or ':', or the parser found an invalid character either in or following the element name.
21	An attribute was not specified correctly. The first character of the attribute name was not a letter, '_', or ':', or a character other than '=' was found following the attribute name, or one of the delimiters of the value was not correct, or an invalid character was found in or following the name.
22	An empty element tag was not terminated by a '>' following the '/'.
23	The element end tag was not specified correctly. The first character was not a letter, '_', or ':', or the tag was not terminated by '>'.

Copyright Central Park Data Systems, Inc

How do you get better at parsing? How do you get to Carnegie Hall?

Google

sample food xml document

About 4,040,000 results (0.27 seconds)

- Everything
- Images
- Videos
- News
- Shopping
- More

Hicksville, NY
Change location

Show search tools

A List Apart: Using XML : Sample XML Document

This file contains a **sample XML** document that describes nutrition facts. ... <protein units="g">50</protein> </daily-values> <food> <name>Avocado Dip</name> ...
www.alistapart.com/d/usingxml/xml_uses_a.html - Cached - Similar

XML Examples

XML and XSLT. View an **XML food** menu - View the corresponding XSLT stylesheet ...
Examples might be simplified to improve reading and basic understanding ...
www.w3schools.com/xml/xml_examples.asp - Cached - Similar

XmlDocument and XmlNode Basics

Once the Xml file is loaded, you can also locate the root node of the document. Source
Example. In my example source code, I load the **Xml Document** using the ...
www.csharpfriends.com/Articles/getArticle.aspx?articleID... - Cached - Similar

Sample Code

There is also a data directory containing a **sample XML document**, DookCatalog.xml, that is used by all the AJAX examples. ...
download.oracle.com/docs/cd/E17802_01/.../docs/.../SAX6.html - Cached

Sample XML Documents (LINK to XML)

The following example files are used in the code samples and code snippets throughout the LINK to XML documentation.
msdn.microsoft.com/en-us/library/bb387026.aspx - Cached - Similar

Sample Soul Food Menu Template - Free Downloads of Sample Soul...

Sample Soul Food Menu Template: flash maker javascript excel builder ... This is a fully customisable CS3 AS2 flash **document** template which rotates a ... editing a general XML settings file and also individual XML configuration files ...
www.freedomdownloadcenter.com/terms/Sample...food.../template.html - Cached



Copyright Central Park Data Systems, Inc

What We'll Cover

- Essential Definitions
- XML – A First Look
- A Review of Parsers
- XML-INTO: A Closer Look
- XML-SAX: A Closer Look
- Wrap-up

Copyright Central Park Data Systems, Inc.

35

Introducing...



XML-SAX !!!

- You DO NOT need to know the structure format in advance.
- Data is mapped anywhere you want at runtime.
- Reads the document the same way we do, left to right from top to bottom.
- Each new piece of encountered data is an event, which is passed to an event handler.
- Recommended for more complex structured documents.

Copyright Central Park Data Systems, Inc.

Using the SAX parser on a complex document

- We have a great deal of EDI programming experience. In that environment it is common to segregate header and detail data and write the values to a database file.
- If we could somehow extract just the data from each XML statement, and write it to a 'real' database file, we would be on our way to processing the data in a format we are more comfortable with.
- We would then have a second program to read this file sequentially (arrival sequence) and process the data without the distraction of the XML parser.

Copyright Central Park Data Systems, Inc.

37

This document contains both XML elements and attributes, and the data is inconsistent. It is however well-formed.

```
<?xml version="1.0" encoding="UTF-8" />
<!DOCTYPE XMLFileIn (View Source for full doctype...)>
<!-- XML ORDER EXPORT FILE V1.0 -->
- <XMLFileIn>
- <OrderV2>
  <Attribute Name="Dealer">10912</Attribute>
  <Attribute Name="Homeowner">000002</Attribute>
  <Attribute Name="ProjectName">New Construction 529</Attribute>
  <Attribute Name="CreatedBy">11714</Attribute>
  <Attribute Name="CustPONumber">Verbal</Attribute>
  <Attribute Name="InhNumber" />
  <Attribute Name="PriceMultiplier">.83</Attribute>
  <Attribute Name="SalesTaxPercent">8.65</Attribute>
  <Attribute Name="SalesTaxAmount">0.00</Attribute>
- <CustomerInfo>
  <Attribute Name="Name" />
  <Attribute Name="CustomerID" />
- <Billing>
  <Attribute Name="Contact" />
  <Attribute Name="Address1" />
  <Attribute Name="Address2" />
  <Attribute Name="City" />
  <Attribute Name="State" />
  <Attribute Name="ZipCode" />
  <Attribute Name="Phone" />
</Billing>
- <Shipping>
  <Attribute Name="Contact" />
  <Attribute Name="Address1" />
  <Attribute Name="Address2" />
```

Copyright Central Park Data Systems, Inc.

38

All of the possible sax event codes:

```

- _QRNU_XMLSAX
  - _QRNU_XMLSAX(1) = ATTR_UCS2_REF
  - _QRNU_XMLSAX(2) = ATTR_NAME
  - _QRNU_XMLSAX(3) = ATTR_PREDEF_REF
  - _QRNU_XMLSAX(4) = ATTR_CHARS
  - _QRNU_XMLSAX(5) = C_CHARS
  - _QRNU_XMLSAX(6) = COMMENT
  - _QRNU_XMLSAX(7) = UCS2_REF
  - _QRNU_XMLSAX(8) = PREDEF_REF
  - _QRNU_XMLSAX(9) = DOCTYPE_DECL
  - _QRNU_XMLSAX(10) = ENCODING_DECL
  - _QRNU_XMLSAX(11) = END_CDATA
  - _QRNU_XMLSAX(12) = END_DOCUMENT
  - _QRNU_XMLSAX(13) = END_ELEMENT
  - _QRNU_XMLSAX(14) = END_PREFIX_MAPPING
  - _QRNU_XMLSAX(15) = EXCEPTION
  - _QRNU_XMLSAX(16) = PI_TARGET
  - _QRNU_XMLSAX(17) = PI_DATA
  - _QRNU_XMLSAX(18) = STANDALONE_DECL
  - _QRNU_XMLSAX(19) = START_CDATA
  - _QRNU_XMLSAX(20) = START_DOCUMENT
  - _QRNU_XMLSAX(21) = START_ELEMENT
  - _QRNU_XMLSAX(22) = START_PREFIX_MAPPING
  - _QRNU_XMLSAX(23) = UNKNOWN_ATTR_REF
  - _QRNU_XMLSAX(24) = UNKNOWN_REF
  - _QRNU_XMLSAX(25) = VERSION_INFO
  - _QRNU_XMLSAX(26) = END_ATTR

```

Copyright Central Park Data Systems, Inc.

40

The sax parser reads the document the same way you read a book; one word at a time, left to right, top to bottom.

```

<?xml version="1.0" encoding="UTF-8" />
<DOCTYPE XMLFileIn (View Source for full doctype...)>
<!-- XML ORDER EXPORT FILE V1.0 -->
<XMLFileIn>
<OrderV2>
<Attribute Name="Dealer">10912</Attribute>
<Attribute Name="Homeowner">000002</Attribute>
<Attribute Name="ParentName">New Generation 500</Attribute>

```

20 = Start_Document	
25 = Version_InfoVersion 1.0	
10 = Encoding_Decl	UTF-8
9 = Doctype_Decl	XMLFileIn
6 = Comment	XML Order Export File V1.0
21 = Start_Element	XMLFileIn
5 = Chars	*blank
21 = Start_Element	OrderV2
5 = Chars	*blank
21 = Start_Element	Attribute
2 = Attr_Name	Name
4 = Attr=Chars	Dealer
26 = End_Attr	Name
5 = Chars	10912
13 = End_Element	Attribute

Copyright Central Park Data Systems, Inc.

40

Captured data file SAXDATA contains only FOUR fields

SAXDATA is populated in program SAXPARSE and processed sequentially in the P.O. creation program.

Line 1	Column 1	Replace
.....	T.Name+++++	Functions+++++
000100	A	R SAXDATA
000200	A	DATA TYPE 20
000300	A	DATA 256
000301	A	DOCPATH 20
000302	A	DOCNAME 20
000303		
000306	A	K DOCPATH
000307	A	K DOCNAME

We already know which pieces of data we want to capture. Each time the event handler is called, if the event code identifies one of these known values, we capture it. Otherwise we simply ignore it.

Copyright Central Park Data Systems, Inc.

41

Program Listing SAXPARSE

```
*SAXPARSE.PRG
Line 38      Column 6      Insert      12 Changes
.....1.....2.....3.....4.....5.....6.....7.....8
000300      Fsaxctl1      uf      e      disk
000400      Fsaxdata      uf      a      k disk
000500
000600      D num_elems      c      500
000700      D xmldoc      s      50      varying
000800      D options      s      20      inz('doc=file')
000900      D currentdate      s      D      inz('xxyy')
001000
001100      // Communications area definition
001200      D myCommArea      ds
001300      D namecount      101 0
001400      D elementdata      dim(num_elems)
001500      D name      64a      overlay(elementData)
001600      D startcount      101 0      overlay(elementData: *next)
001700      D endcount      101 0      overlay(elementData: *next)
001800
001900
002000      D EventHandler      pr      101 0
002100      D commArea      likeds(myCommArea)
002200      D event      101 0      value
002300      D pstring      *      value
002400      D stringlen      201 0      value
002500      D exceptionID      101 0      value
002600
002700      /free
002800
002900      // Read unprocessed records only (field prcflg = 'blank')
003000      read saxctl1:
003100      dow not %eof (saxctl1);
003200
003300      // Delete any records in output file with same path and document name
003400      // If any are found, they are from a previous run
003500      dou not %found(saxdata);
003600      delete(e) (docpath:docname) saxdata:
003700      enddo;
003800
003900      // Build the document path into field "xmldoc"
004000      // The document folder will always be in the root directory
```

Copyright Central Park Data Systems, Inc.

42

*XMLSAX debugging option

```
h dftactgrp(*no) actgrp(*caller) debug(*xmlsax)
Fsaxdata o a e disk
```

- The *XMLSAX debug option generates a special array named **_QRNU_XMLSAX** into your program. Since the SAX parser returns just a numeric value, it is helpful to identify what each value represents.
- In the event handler we can test for each event and take the appropriate action. We will capture certain pieces of data, based on the event code that is passed.

43

Copyright Central Park Data Systems, Inc.

This is how the parsed data looks in file SAXDATA.
A second program will read these records sequentially and process the data.

Event Codes	Start_Element	XMLFileIn
21	Start_Element	OrderV2
2	Start_Element	Attribute
4	Attr_Name	Name
4	Attr_Chars	Dealer
5	XML_Chars	10912
13	End_Element	Attribute
	Start_Element	Attribute
	Attr_Name	Name
	Attr_Chars	Homeowner
	XML_Chars	000002
	End_Element	Attribute
	Start_Element	Attribute
	Attr_Name	Name
	Attr_Chars	ProjectName
	Start_Element	FinishedGood
	Start_Element	PartNumber
	XML_Chars	1631516
	End_Element	PartNumber
	Start_Element	Attribute
	Attr_Name	Name
	Attr_Chars	Order_Quantity
	XML_Chars	4
	End_Element	Attribute
	Start_Element	Attribute
	Attr_Name	Name
	Attr_Chars	MfgNumber
	XML_Chars	HG-5522GH
	End_Element	Attribute
	Start_Element	Attribute

44

Copyright Central Park Data Systems, Inc.

Code snippet of how to process this captured sequential data

```
If datatype = 'Attr_Chars';  
Select;  
When data = 'Dealer';  
    flag = 'Dealer';  
...
```

> Sets flag

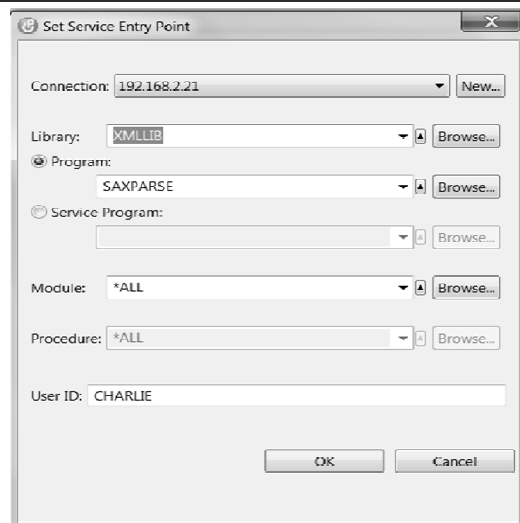
```
If datatype = 'XML_Chars';  
Select;  
When flag = 'Dealer';  
    dealerno = %trim(data);  
When flag = 'Homeowner';  
    homeownerdata = %trim(data);  
...
```

> Map suitable data

Copyright Central Park Data Systems, Inc.

45

LIVE DEMONSTRATION OF SAXPARSE



Here we are setting the Service Entry Point, or SEP

Copyright Central Park Data Systems, Inc.

46

Screen shot of file SAXCTL

```
WORK WITH DATA IN A FILE                               Mode . . . . :  CHANGE
Format . . . . :  SAXCTL                               File . . . . :  SAXCTL

*RECNBR:          1
DOCPATH: xml docs
DOCNAME: sample2.xml
PRCFLG:  _
PRCDAT: 0001-01-01
PRCTIM: 01.01.01
```

Logical file SAXCTL1 selects only records that contain '0001-01-01' in the processed date field "PRCDAT".

Copyright Central Park Data Systems, Inc.

47

Read SAXCTL file to see if any files exist to be parsed

```
// Read unprocessed records only (field prctlg = *blank)
read saxctl1;
do not %eof (saxctl1);

// Delete any records in output file with same path and document name
// If any are found, they are from a previous run
do not %found(saxdata);
  delete(e) (docpath:docname) saxdata;
enddo;

// Build the document path into field "xml doc"
// The document folder will always be in the root directory
clear xml doc;
xml doc = '/' + %trim(docpath) + '/' + %trim(docname);
```

Copyright Central Park Data Systems, Inc.

48

Here file “xmldocs/sample2.xml” in the root directory will be parsed

```
// Build the document path into field "xmldoc"
// The document folder will always be in the root directory
clear xmldoc;
xmldoc = '/' + %trim(docpath) + '/' + %trim(docname);
XMLDOC = /xmldocs/sample2.xml
// If an error occurs, mark the process flag with an 'E'
monitor;
// Begin the parsing, processing is similar to a subroutine.
// Control will be passed to the "handler" for each event encountered.
```

Copyright Central Park Data Systems, Inc.

49

First time in handler, event code = 20 – OR – “*XML START DOCUMENT”

```
select;
// Communications area definition
when event = *XML START DOCUMENT;
clear EVENT = 20;
when event = *XML START ELEMENT;
name = %subst(string : 1 : stringlen);
clear *nokey saxdata;
datatype = 'Start_Element';
data = name;
write saxdata;
when event = *XML END_ELEMENT;
name = %subst(string : 1 : stringlen);
clear *nokey saxdata;
datatype = 'End_Element';
data = name;
write saxdata;
when event = *XML ATTR NAME;
name = %subst(string : 1 : stringlen);
clear *nokey saxdata;
datatype = 'Attr_Name';
data = name;
write saxdata;
```

Copyright Central Park Data Systems, Inc.

50

Second time in handler, event code = 25 – OR – “*XML_VERSION_INFO”

```

/free

select;

// Communications area definition
when event = *XML_START_DOCUMENT:
  clear commArea;

when event = *XML_START_ELEMENT:
  name = %EVENT = 25 : 1 : stringlen);
  clear *nokey saxdata;
  datatype = 'Start_Element';
  data = name;
  write saxdata;

when event = *XML_END_ELEMENT:
  name = %subst(string : 1 : stringlen);
  clear *nokey saxdata;
  datatype = 'End_Element';
  data = name;
  write saxdata;

```

Copyright Central Park Data Systems, Inc.

51

Parsed data shown in file SAXDATA using Table View

DATA TYPE	DATA	DOC...	DOC...
Start_Element	XMLFileIn	xmldocs	sample2...
Start_Element	OrderV2	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...
Attr_Chars	Dealer	xmldocs	sample2...
XML_Chars	10912	xmldocs	sample2...
End_Element	Attribute	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...
Attr_Chars	Homeowner	xmldocs	sample2...
XML_Chars	000002	xmldocs	sample2...
End_Element	Attribute	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...
Attr_Chars	ProjectName	xmldocs	sample2...
XML_Chars	New Construction 529	xmldocs	sample2...
End_Element	Attribute	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...
Attr_Chars	CreatedBy	xmldocs	sample2...
XML_Chars	11714	xmldocs	sample2...
End_Element	Attribute	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...
Attr_Chars	CustPONumber	xmldocs	sample2...
XML_Chars	Verbal	xmldocs	sample2...
End_Element	Attribute	xmldocs	sample2...
Start_Element	Attribute	xmldocs	sample2...
Attr_Name	Name	xmldocs	sample2...

Copyright Central Park Data Systems, Inc.

52

What We'll Cover

- **Essential Definitions**
- **XML – A First Look**
- **A Review of Parsers**
- **XML-INTO: A Closer Look**
- **XML-SAX: A Closer Look**
- **Wrap-up**

Copyright Central Park Data Systems, Inc.

53

Wrap-up

- XML-INTO is appropriate for less complex complicated structures.
 - XML parsing was first introduced into RPG at V5R4.
 - It parses the XML data into data structures, therefore you **MUST** know the names of the tags in advance.
 - If you are missing tags – or – define too many tags in your data structure and do not specify “allow-missing” or “allow-extra” you will receive an RPG parsing error at run-time.
 - PTF SI34938, which became available in early 2009 enhanced XML-INTO, allowing it the ability to parse more complex XML structures.

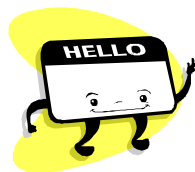
Copyright Central Park Data Systems, Inc.

54

Wrap-up (continued)

- XML-SAX is appropriate for any type of XML structure.
 - It is an event driven parser, which reads your document the same way you read a regular document – left to right, top to bottom.
 - An event handler is called for each event that is encountered.
 - ▶ Along with each event is an event code.
 - “Allow-missing” and “allow-extra” are meaningless to the SAX parser since you do not pre-define your tags.
 - You do not need to handle or capture the event code every time the handler is called. Only capture what you need.

Looking Inside the Developer's Toolkit: Introduction to XML and RPG



Charles Guarino

THANK YOU !!!

Sessions by Charles Guarino

Sunday 9:30AM	REST Web Services for Everyday RPG Consumption Using REST to extend your applications beyond your four walls
Sunday 12:30PM	Raising the Bar: A Jumpstart to Using SOAP Web Services in RPG A step-by-step discussion on implementing IBM's WSDL2RPG
Monday 11:00AM	From STRPDM to RDP An introduction to RDP and source code editing
Monday 2:00PM	From STRPDM to RDP – The Journey Continues Exploring advanced features in RDP and making it your own
Monday 3:30PM	From STRSDA and STRRLU to RDP A step by step demonstration on using the RDP visual designer
Tuesday 9:30 AM	From STRDBG to the RDP Debugger A walk through on using the debugging perspective
Wednesday 9:30AM	Introduction to Processing XML in RPG A beginner's guide to processing XML in RPG
Wednesday 2:00PM	There's Gold in Them Thar Hills! Making the most of an RPG formatted dump
Wednesday 5:00PM	Program Application Recovery and Other Tricks Proven techniques to keep your cool before and after an IT crisis

Copyright Central Park Data Systems, Inc.

57