

Intro to XML

Mark Shurr

Ada Business Technology Inc.

March 17, 2010

Agenda

- About Mark Shurr
- You probably know XML basics already
 - HTML
 - XHTML
 - XML Samples
- Brief History of XML
- Basic XML Format
- XML Terminology
- What is a DTD (it may sound familiar to those of us who still remember COBB)
- Editing XML and DTD Files

Agenda Continued

- Almost transparent communications over the Internet
 - XML and Web Services (SOAP)
 - Sample Web Service with SOAP XML
- Popular XML Tools
 - SAX
 - DOM
 - IBM Java XML
 - .NET Parsing
- XML Parsing on the IBM i
- XML Toolkit for iSeries (5733XT1)
 - XML-SAX
 - XML-INTO
- Parsing XML in .NET
- Commercial Tools that make our life easy
 - LANSAX
 - XML/400
 - RPG-XML Suite
- References

About Mark Shurr

- Mark is the Vice President and Principal Consultant of Ada Business Technology, Inc.
- He has over 20 years of experience in the development of business and manufacturing systems in the AS400/System i and Microsoft environment. He is proficient in RPG/RPGLE as well as VB and ASP.NET. He has designed and implemented several Database Systems including: IBM DB/2, Microsoft SQL Server 2005 /2008 and MySQL.

You probably know XML basics already 1/3

- HTML- Hyper Text Markup language

```
<HEAD>
<title>Basic HTML Sample Page</title>
</HEAD>
<body bgcolor="white">
<center>
<TABLE border cellspacing=4 cellpadding=14>
<TR><TD valign=top width="48%"> </TD></TR>
<center> This is a center </center>
</center>
</body>
```

You probably know XML basics already 2/3

- XHTML - Extensible Hyper Text Markup language

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"DTD/xhtml1-frameset.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title> Frameset DTD XHTML Example </title>
</head>
<frameset cols="100,*">
<frame src="toc.html" />
<frame src="intro.html" name="content" />
</frameset>
</html>
```

You probably know XML basics already 3/3

- XML Samples – Extensible Markup Language

```
<?xml version="1.0"?>
<catalog>
  <book id="bk101">
    <author>Gambardella, Matthew</author>
    <title>XML Developer's Guide</title>
    <genre>Computer</genre>
    <price>44.95</price>
    <publish_date>2000-10-01</publish_date>
    <description>An in-depth look at creating applications
with XML.</description>
  </book>
</catalog>
```

Brief History of XML

- Origins in SGML in the 1970's – Standard Generalized Markup Language by Dr Charles Goldfarb
- In the late 1990s a group of people including Jon Bosak, Tim Bray, James Clark and others came up with XML, eXtensible Markup Language
- In the 1990's SUN and IBM adopted XML
- Microsoft picked on XML as an alternative approach to the interoperability puzzle, and became XML's greatest advocate

Basic XML Format 1/3

An XML document is a tree of nested elements, each of which can have none or more attributes. There can only be one root element. Each element has an starting and ending tag, marked by angle brackets, with content in between:

```
<element>...content...</element>
```

The content can contain other elements, or can consist entirely of other elements, or might be empty.

Attributes are named values which are given in the start tag, with the values surrounded by single or double quotations:

```
<element attribute1="value1" attribute2="value2">
```

Note an element can be contained in a single line

```
<element attribute 1 />
```

Basic XML Format 2/3

Declaration

This line declares that what follows is an XML document and specifies the version, currently always 1.0, and optionally the encoding or character set, and whether the documents stands alone or requires an external schema file. Example:

```
<?xml version="1.0" encoding="utf-8" ?>
```

Another type of declaration indicates the document type, either by reference to an external DTD or by including it inline:

```
<!DOCTYPE pcwdoc SYSTEM "http://itwriting.com/itwriting.dtd" >
```

Comment

XML comments appear between special delimiters:

```
<-- a comment -->
```

Basic XML Format 3/3

Elements and attributes

The basic building blocks of XML. Elements are hierarchical, must have start and end tags, and optionally include attributes in the start tag. Empty elements are those that have no content, and may use a combined start and end tag like `<hr/>`. An example element:

```
<magazine title="Personal Computer World"> ... </magazine>
```

Character data

Plain text that forms the content of elements. Elements can also include other elements.

XML Terminology 1/2

- DOM: Document Object Model. This is a way of representing an XML document as a hierarchy of objects that can be manipulated programmatically (can randomly pick elements)
- DTD: Document Type Definition. A document which defines elements, attributes and other constraints so that XML documents of the specified type can be validated. An alternative to DTDs are XML Schema
- EXPAT An open source XML parser written in C. It is know for its effective parsing speed. It's a stream-oriented parser, similar in many ways to SAX
- SAX: Simple API For XML. Originally a JAVA API for parsing XML. It works by raising events as each significant item in the document being parsed (one time top to bottom parser)
- SOAP: Simple Object Access Protocol. An XML protocol for doing application messaging and remote procedure calls over the internet

XML Terminology 2/2

- UDDI: Universal Description, Discovery and Integration. A public registry that enables businesses to publish information about the web services they offer. UDDI offers a programmatic interface as well as a user interface
- Web Service: An application accessible over the Internet through an XML interface
- WSDL: Web Service Description Language. An XML format for describing XML web services, including the type definitions, messages and actions used by that service. The WSDL document should tell applications all they need to know to invoke a particular web service
- XSD: XML Schema Definition. A document containing an XML Schema, with an .xsd extension
- XQUERY: XML Query Language. Work in progress to define a language for querying one or more XML documents, a sort of XML equivalent to SQL (Structured Query Language). XPATH, XPOINTER, XQUERY and XSLT have considerable overlap, a fact which is recognized by the various working groups

Almost transparent communications over the Internet 1/3

- XML and Web Services (SOAP)
 - Sample Web Service with SOAP XML

```
POST /EarnedValue.asmx HTTP/1.1
Host: www.adabustech.com
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://AdaBusTech.com/EV"
```

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <EV xmlns="http://AdaBusTech.com/">
      <BCWS>int</BCWS>
      <BCWP>int</BCWP>
      <ACWP>int</ACWP>
    </EV>
  </soap:Body>
</soap:Envelope>
```

Almost transparent communications over the Internet 2/3

- **Sample Soap Return XML**

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <EVResponse xmlns="http://AdaBusTech.com/">
      <EVResult>
        <EV>int</EV>
        <PV>int</PV>
        <AC>int</AC>
        <CV>decimal</CV>
        <CVP>decimal</CVP>
        <CPI>decimal</CPI>
        <SV>decimal</SV>
        <SVP>decimal</SVP>
        <SPI>decimal</SPI>
        <Errcode>int</Errcode>
      </EVResult>
    </EVResponse>
```


What is a DTD (it may sound familiar to those of us who still remember COBB) 1/3

- A DTD- **Document Type Definition** is a schema that provides structure and logic to an XML file. This is a major difference between CSV & HTML files to XML. XML editors and parsers validate the XML against the DTD.

- Major components of a DTD

- XML Declaration

- `<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>`

What is a DTD (it may sound familiar to those of us who still remember COBB) 2/3

- Document type declaration
 - `<!DOCTYPE Root-Element SYSTEM "Root-Element.dtd">`
(Where the DTD is)
- Element type declaration
 - `<!ELEMENT NAME (#PCDATA)>` (Element type)
 - #PCDATA refers to Text in XML
 - Elements are separate by a delimiter that describes their use
 - `<!ELEMENT CUSTOMER-MASTER (CUSTOMER -NAME @CUSTOMER-NUMBER)>`
 - , Must be used in order shown
 - | One or other element may be used
 - blank Element is required
 - + Must be used at least once
 - * May be used many time as needed or not at all
 - ? May be used once or not at all

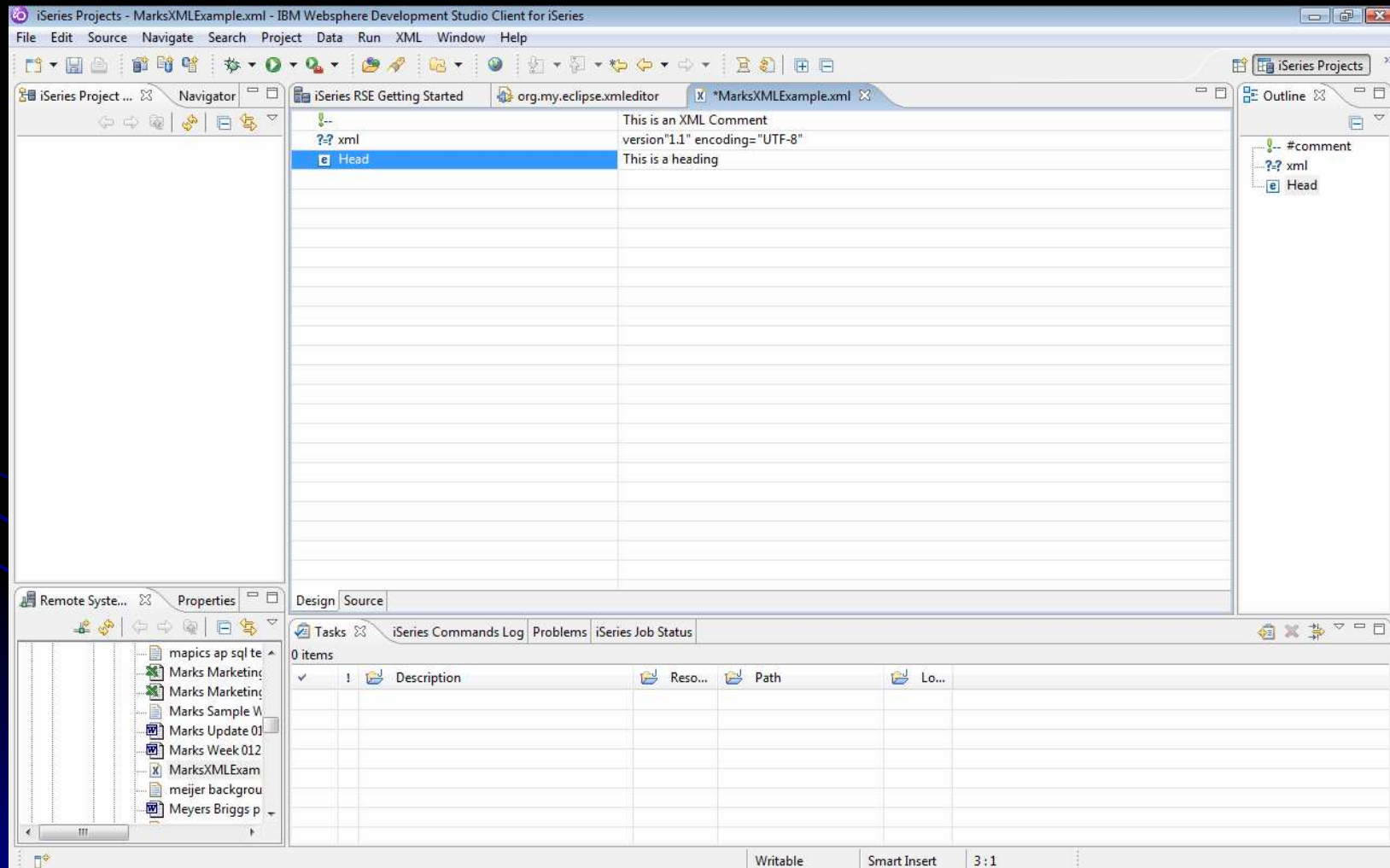
What is a DTD (it may sound familiar to those of us who still remember COBB) 3/3

- Attribute list declaration
 - `<!ATTLIST Element-Name Name Data-Type Default>`
(Defines Attributes associated with the elements)
 - Attributes can be listed with requirements
 - `<ATTLIST CUSTOMER-MASTER CUSTOMER-NAME CDATA #REQUIRED>`
 - #REQUIRED Must always include the attribute
 - #IMPLIED May be used
 - #FIXED Attribute is option but if used must always take default value
 - Value Provides the value to use if nothing else is entered
- Comments
 - `<!-- Include Comments here -->`

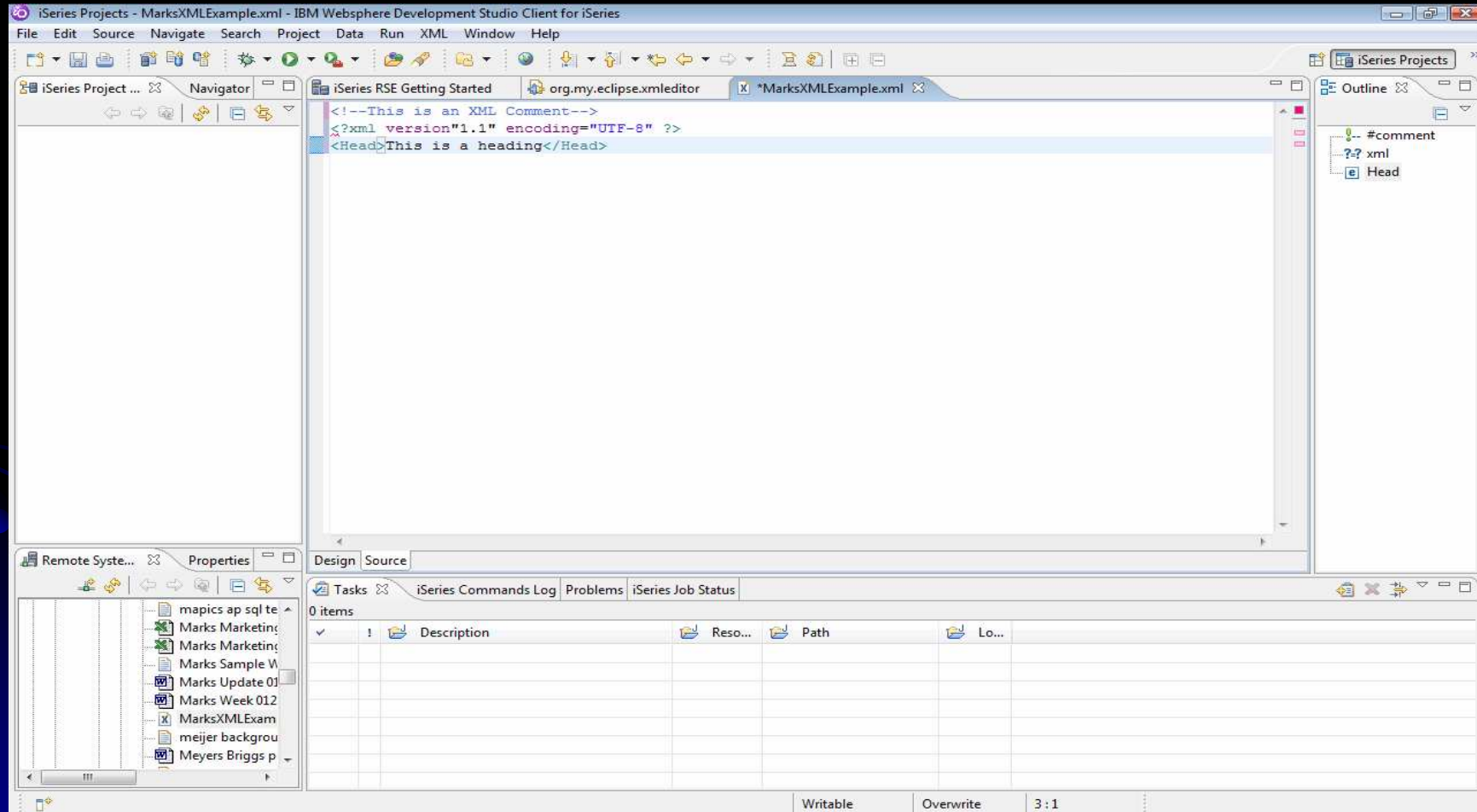
Editing XML and DTD Files 1/3

- XML Editors can be a great asset in creating and changing XML Files
- There are numerous free or almost free editors available today
 - Arbor Text – Adept
 - Adobe – FrameMaker
 - SoftQuad Software - XMETAL
 - Eclipse – IBM has there own basic XML editor that can be installed as a plug in into Websphere Development Studio
 - Follow instructions on the following link
 - <http://www.ibm.com/developerworks/library/os-ecxml/#2>

Editing XML and DTD Files 2/3



Editing XML and DTD Files 3/3



XML Parsing on the IBM i (I still call it the AS400)

- XML Toolkit for iSeries (5733XT1)

- XML-SAX

- XML-SAX uses the API for XML to identify events--start document, end document, start element, end element, characters, etc.

- When SAX finds an event, it notifies a procedure, which must determine what the event is and how to handle it

- `xml-sax %handler(your-handler-here : your-parameter-here) %XML(ifs path name : 'doc=file');`

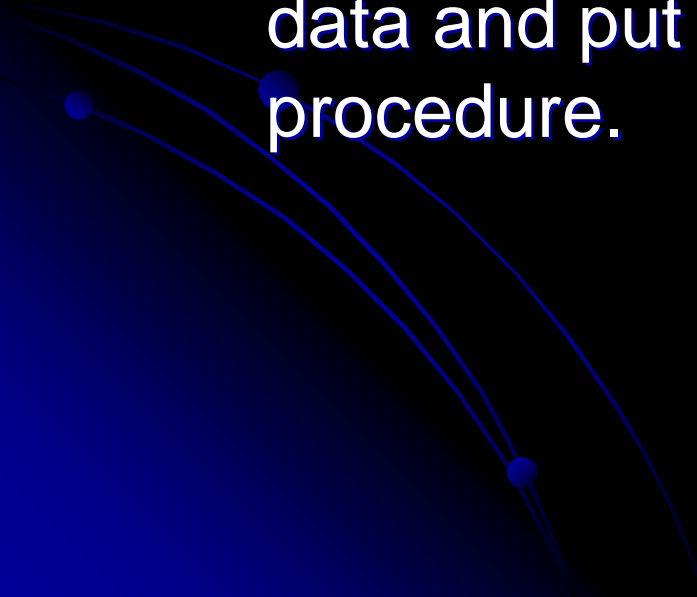
XML Parsing on the IBM i

- XML-SAX Events
 - XML_START_DOCUMENT
 - XML_START_ELEMENT xmlTest
 - XML_START_ELEMENT name
 - XML_ATTR_NAME type
 - XML_ATTR_CHARS author
 - XML_END_ATTR
 - XML_CHARS Scott Klement
 - XML_END_ELEMENT name
 - XML_END_ELEMENT xmlTest
 - XML_END_DOCUMENT

For a good example:

http://publib.boulder.ibm.com/infocenter/iserics/v5r4/index.jsp?to pic=/books_web/c0925086778.htm

XML Parsing on the IBM i (I still call it the AS400)

- XML-INTO, on the other hand, already knows what sort of data it is looking for. That is, you can tell XML-INTO to get a name, an address, and a phone number, and it will pick out the data and put it into variables or feed it to a procedure.
- 

Sample XML - INTO

- Sample XML input

```
<?xml version="1.0" encoding="UTF-8" ?>
- <RequestforUICGroup SchemaVersionMajor="2008-2009" SchemaVersionMinor="1"
  CollectionId="1" CollectionName="RequestforUIC" SubmittingSystemVendor=
  "Computer Management Technologies" SubmittingSystemName=
  "CIMS Student Management System" SubmittingSystemVersion=
  "SMS 0806" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation=
  "https://XXXX.state.XX.us/srsd/xmlschema/UICRequest/RequestforUIC2008-2009.xsd">
- <RequestforUIC>
- <SubmittingEntity>
  <SubmittingEntityTypeCode>D</SubmittingEntityTypeCode>
  <SubmittingEntityCode>09999</SubmittingEntityCode>
</SubmittingEntity>
- <CoreFields>
  <LastName>DUCK</LastName>
  <FirstName>DONALD</FirstName>
  <Gender>F</Gender>
  <DateOfBirth>2003-06-26</DateOfBirth>
  <UIC>9999999999</UIC>
</CoreFields>
```

XML-INTO

- Sample RPGLE Program

```
H DFTACTGRP(*NO)
```

```
D RequestForUIC ds qualified
D dim(999)
D CoreFields liked(Corefields_t)
D EntityDemographics...
D liked(EntityDemographics_t)
```

```
D corefields_t DS qualified
D based(Template)
D LastName 25a
D FirstName 15a
D UIC 10a
```

```
D EntityDemographics_t...
D ds qualified
D based(Template)
D studentIdNumber...
D 9a
```

```
D x s 10i 0
D xmlfile s 1000a varying
D options s 100a varying
D divider s 52a
```

XML-INTO

/free

// CHANGE THIS:

```
xmlfile = '/my/ifs/path/to/students.xml';
```

```
options = 'doc=file +  
          path=RequestForUICGroup/RequestForUIC +  
          case=any +  
          allowextra=yes +  
          allowmissing=yes';
```

```
xml-into RequestForUIC %xml(xmlfile: options);
```



XML-INTO

```
// -----  
// the following is for test/debug purposes only.  
// it just displays the students on the screen.  
// -----  
  
x = 1;  
dow x <= %elem(RequestforUIC)  
  and RequestForUIC(x).Corefields.UIC<>*blanks;  
  divider = *ALL'-';  
  divider = 'Student ' + %char(x) + divider;  
  dsply divider;  
  dsply ( 'Name: '  
    + %trimr(RequestForUIC(x).corefields.LastName)  
    + ', ' + RequestForUIC(x).corefields.FirstName );  
  dsply ( 'UIC: ' + RequestForUIC(x).corefields.UIC );  
  dsply ( 'StudentIdNumber: '  
    + RequestForUIC(x).EntityDemographics.StudentIdNumber );  
  x = x + 1;  
enddo;  
  
*inlr = *on;  
  
/end-free
```

Parsing XML in .NET

```
Imports System.IO
Imports System.Xml
Module ParsingUsingXmlTextReader
Sub Main()
    Dim m_xmlr As XmlTextReader
    'Create the XML Reader
    m_xmlr = New XmlTextReader("C:\Personal\family.xml")
    'Disable whitespace so that you don't have to read over whitespaces
    m_xmlr.WhiteSpaceHandling = WhiteSpaceHandling.NONE
    'read the xml declaration and advance to family tag
    m_xmlr.Read()
    'read the family tag
    m_xmlr.Read()
    'Load the Loop
    While Not m_xmlr.EOF
        'Go to the name tag
        m_xmlr.Read()
        'if not start element exit while loop
        If Not m_xmlr.IsStartElement() Then
            Exit While
        End If
        'Get the Gender Attribute Value
        Dim genderAttribute = m_xmlr.GetAttribute("gender")
        'Read elements firstname and lastname
        m_xmlr.Read()
        'Get the firstName Element Value
        Dim firstNameValue = m_xmlr.ReadElementString("firstname")
        'Get the lastName Element Value
        Dim lastNameValue = m_xmlr.ReadElementString("lastname")
        'Write Result to the Console
        Console.WriteLine("Gender: " & genderAttribute _
            & " FirstName: " & firstNameValue & " LastName: " _
            & lastNameValue)
        Console.WriteLine(vbCrLf)
    End While
    'close the reader
    m_xmlr.Close()
End Sub
End Module
```

Commercial Tools that make our life easy

1/3

- LANSa – Integrator
 - Enables integration of Application-to-Application (A2A) and Business-to-Business (B2B) transactions through XML and Java services. LANSa Integrator allows bi-directional XML – and other data formats – to be exchanged between host and your trading partners, regardless of platform. It also enables integration of user-written Java services with LANSa, C, RPG and COBOL applications.

Commercial Tools that make our life easy

2/3

- **ALIANCE/400 - Patrick Townsend Security Solutions**
 - Provides a complete secure web services solution for IBM System i. With Alliance XML/400 the user can automatically translate XML documents directly to a System i DB2 database files. When the user wants to create and send XML documents, Alliance reads DB2 files and builds an XML document. Then the user chooses the communications protocols to deliver the XML data, including HTTP or HTTPS web communications, Microsoft SharePoint web folders, File Transfer Protocol (FTP), or Websphere MQ (MQSeries).

Commercial Tools that make our life easy

3/3

- RPG-XML Suite By Kringel Tech
 - Parses, composes, and transmits XML
 - Offers a web service on the System i
 - Calls or consumes a web service on another machine
 - Fully supports HTTPS/SSL
 - Compatible with SOAP-based web services
 - 100% System i native toolset!
 - Creates Microsoft Word and Excel documents using XML.
 - Code generators allow programs to be built FAST - Lessening the typical learning curve and time required to build programs from scratch.
 - No Java. No Websphere. 100% RPG. There are NO additional technologies, hardware, or software required to effectively implement RPG-XML Suite to transmit XML.

References

- XML for Dummies By Frank Boumphrey
- Wikipedia
- "Real World" Example of XML-INTO “ By Scott Klement – System I Network & Mid Range Magazine articles
- IBM XML-INTO
http://publib.boulder.ibm.com/infocenter/iseries/v5r4/index.jsp?topic=/books_web/c0925086773.htm

Ada Business Technology, Inc.

- Specializes in:
 - IBM Midrange Systems Design & Implementation
 - Business & Manufacturing System (ERP, CRM)
 - IT Governance
 - Project Management
- For additional support please contact Mark Shurr @

Ada Business Technology Inc.

mshurr@AdaBusTech.com

<http://www.AdaBusTech.com>

PO Box 734, Ada MI 49301

Office: 616.805.7402

Cell: 631.742.5178