JSchema Users & Tools Guide
Ver. 2.3

Introduction

Welcome to I-Technologies Corp's JSchema and XML family of open source projects.

What is JSchema?

JSchema is a Java framework and API for reading and writing XML Schema documents. In addition, it has code generation tools for generating Java classes and SQL tables from XML schema documents and vice versa. JSchema uses the XML Java marshaling tools in the ItcWorks open source project (also created by ITC). These XML parsing tools use SAX parsers that provide the fastest possible execution speeds.

The framework consists of:

Java interface classes (one interface for each XML Schema type) a set of implementation classes, a schema reader (which takes a URL to a schema document and produces schema object) and a schema writer (that takes a schema object and produces a schema document at the specified location). JSchema is based on the W3cs' XML Schema 1.0 specification. XML schemas have become ideal for representing object oriented classes in a language neutral way.

Each schema xml element has its own Java interface. (i.e., <complexType> has a ComplexType interface). The attributes of a given XML schema tag are represented as properties in the interface. Additional helper methods exist for many of the interface classes for searching, adding or modifying child components. For example, the ComplexType element has methods like hasAttributes() (which determine if type has *attributes* defined) or hasModelGroup() (which determines if the complexType is composed of *sequence*, *choice* or *all* child elements). JSchema also supports user defined attributes that can be added to any of the existing XML schema elements.

Audience:

JSchema is for Java developers that need to incorporate XML schema data into their applications. Its rich API makes it simple to read, modify, or create XML schema documents. JSchema is ideal for tool developers building GUI front ends for schema manipulation. XML Schemas also make great data dictionaries for field validation from user input forms.

What is in JSchema?

JSchema is organized into the following packages:

- javax.xml.schema – This package contains the interface classes that define the complete set schema elements.
- javax.xml.namespace – This package contains two concrete classes: Namespace and QName.
- javax.xml.schema.util – This package contains factory classes for creating the JSchema reader and writer objects.
- com.itc.xml.schema - implementation classes for the javax.xml.schema package
- com.itc.xml.util - implementation classes for the javax.xml.schema.util package

 Copyright © 2006, I Technologies Corp

- com.itc.xml.schema.tools -code generation tools

JSchema is packaged in itcworks.jar. The following jars are also required:

- xercesImpl.jar – xml parsing
- xercesAPIs.jar – xml parsing

Reading XML Schema documents:

The following is a code snippet that shows how to read XML schema documents using JSchema:

```
File fileSchema = new File( "\\purchaseOrder.xsd");

Schema schema = SchemaFactory.getInstance().newReader().readSchema( fileSchema.toURL()
);

// Get a List of all complex types found in the schema
List cmplxTypes = schema.getComplexTypes();
```

Writing XML Schema documents:

The following is a code snippet that shows how to write XML schema documents using JSchema:

```
File fileSchema = new File( "\\purchaseOrder.xsd");
SchemaFactory.getInstance().newWriter().writeSchema( schema, fileSchema );

// Or write schema to a string
String strSchema = SchemaFactory.getInstance().newWriter().writeSchema( schema );
```

Schema Tools

ItcSchemaToJava

The ItcSchemaToJava tool generates Java object graphs (data value objects (DVO's)) from complexType tags and elements that have anonymous complexType tags. This tool can also be run from the command line. The command line options are described below:

-f (required) This identifies the input XML schema definition file i.e. /myschemas/myschema.xsd
-o (required) The output directory where the classes will be written. This is the path to the start of the output directory
-p (required unless –t option is used) This is the package name for the generated Java classes
-t (required unless –p option is used) Use the target namespace as the package name
-a attribute normal form. This option specifies that the attributes of each element are generated as the properties and each element is generated as the class. The default is to generate the complexType name as the class and the child elements as the properties.
-u (optional) generate object types for java primitives i.e., Integer instead of int ….

 Copyright © 2006, I Technologies Corp

-r complexTypeName (optional) Generates a reader and writer class for a schema document, where ***complexTypeName*** is the schema type that represents the top level class.
Note! It is recommended this option is always used.

-m This sets a property in the reader class (if the –r option is used) that turns off macro expansion. Macro expansion is defined by using the ${propertyName} format as attributes or element values in the schema instance document. By default, these values translated when the XML instance document is read into the object graph.

Examples:
Given the following schema:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:tns="http://mycompany.com/dvo"
            targetNamespace="http://mycompany.com/dvo">

  <xsd:element name="employees" type="tns:Employees"/>
  <xsd:complexType name="Employees">
    <xsd:sequence>
      <xsd:element name="employee" type="tns:Employee"
                   maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="Employee">
    <xsd:sequence>
      <xsd:element name="address" type="tns:Address"/>
     </xsd:sequence>
     <xsd:attribute name="name" type="xsd:string"/>
         <xsd:attribute name="dept" type="xsd:string"/>
      <xsd:attribute name="yearsEmployeed" type="xsd:int"/>
  </xsd:complexType>
  <xsd:complexType name="Address">
      <xsd:attribute name="stree1" type="xsd:string"/>
  </xsd:complexType>

</xsd:schema>
```

java com.itc.xml.schema.tools.ItcSchemaToJava –f /myschemas/employees.xsd –o /MyWorkspace/MyProject/src –t –u –r Employees

The above example reads the employees.xsd document and generates the package folder com/mycompany/dvo starting at the absolute location of /MyWorkspace/MyProject/src. The ***–t*** specifies the package names will be derived from the target namespace defined the schema document. The ***–u*** generates Java object types instead of primitives for the DVO's generated. The ***–r Employees*** generates reader and writer classes (for the instance xml documents) into the top level object Employees.

Running this example produces the following files:

`Employees.java`

```
/*
================================================================================
===

                    I t c W o r k s   C o d e   G e n e r a t o r

                         2001-2007 by i Technologies Corp
```

```
    Source File Name: Employees.java

    Author:

    Date Generated:   08-19-2007

    Time Generated:   07:13:56

================================================================================
===
*/

package com.mycompany.dvo;

import java.util.List;


public class Employees
{

  private List<Employee>         m_listEmployee;

  // *** The following members set or get data from the class members ***

  /**
   * Sets the employee property

   */
  public void setEmployee( List<Employee> listEmployee )
  { m_listEmployee = listEmployee; }

  /**
   * Gets employee property
   *
   * @return  The employee property
   */
  public List<Employee> getEmployee()
  { return m_listEmployee; }
} // *** End of class Employees{}

// *** End Of Employees.java
```

Employee.java

```
/*
================================================================================
===

                        I t c W o r k s   C o d e   G e n e r a t o r

                              2001-2007 by i Technologies Corp

    Source File Name: Employee.java

    Author:

    Date Generated:   08-19-2007

    Time Generated:   07:13:56

================================================================================
===
*/

package com.mycompany.dvo;
```

```
public class Employee
{

  private String                m_strName;
  private String                m_strDept;
  private Integer               m_yearsEmployeed;
  private Address               m_address;

  // *** The following members set or get data from the class members ***

  /**
   * Sets the name property

   */
  public void setName( String strName )
  { m_strName = strName; }

  /**
   * Gets name property
   *
   * @return  The name property
   */
  public String getName()
  { return m_strName; }

  /**
   * Sets the dept property

   */
  public void setDept( String strDept )
  { m_strDept = strDept; }

  /**
   * Gets dept property
   *
   * @return  The dept property
   */
  public String getDept()
  { return m_strDept; }

  /**
   * Sets the yearsEmployeed property

   */
  public void setYearsEmployeed( Integer yearsEmployeed )
  { m_yearsEmployeed = yearsEmployeed; }

  /**
   * Gets yearsEmployeed property
   *
   * @return  The yearsEmployeed property
   */
  public Integer getYearsEmployeed()
  { return m_yearsEmployeed; }

  /**
   * Sets the address property

   */
  public void setAddress( Address address )
  { m_address = address; }

  /**
   * Gets address property
   *
   * @return  The address property
```

 Copyright © 2006, I Technologies Corp

```
   */
  public Address getAddress()
  { return m_address; }
} // *** End of class Employee{}

// *** End Of Employee.java
```

Address.java

```
/*
================================================================================================
===

                        I t c W o r k s   C o d e   G e n e r a t o r

                              2001-2007 by i Technologies Corp

    Source File Name: Address.java

    Author:

    Date Generated:   08-19-2007

    Time Generated:   07:13:56

================================================================================================
===
*/

package com.mycompany.dvo;



public class Address
{

  private String                m_strStree1;

  // *** The following members set or get data from the class members ***

  /**
   * Sets the stree1 property

   */
  public void setStree1( String strStree1 )
  { m_strStree1 = strStree1; }

  /**
   * Gets stree1 property
   *
   * @return  The stree1 property
   */
  public String getStree1()
  { return m_strStree1; }
} // *** End of class Address{}

// *** End Of Address.java
```

EmployeesReader.java

```
/*
================================================================================================
===

                        I t c W o r k s   C o d e   G e n e r a t o r
```

```
                                2001-2007 by i Technologies Corp

    Source File Name: EmployeesReader.java

    Author:

    Date Generated:   08-19-2007

    Time Generated:   07:13:56

================================================================================================
===
*/

package com.mycompany.dvo;

import java.net.URL;
import org.xml.sax.InputSource;
import com.itc.util.ItcResourceStoreFactory;
import com.itc.xml.ItcXmlToBean;
import javax.xml.schema.util.XmlDeSerializer;


public class EmployeesReader
{


  /**
   * Reader

   */
  public static Employees read( URL urlDoc ) throws Exception
  {

    URL urlSchemaXSD = ItcResourceStoreFactory.getInstance().getStore().getDocument(
"employees.xsd" );

    ItcXmlToBean xtb = new ItcXmlToBean();

    xtb.setFeature( XmlDeSerializer.ATTRIBUTE_MODEL, true  );
    return (Employees) xtb.deSerialize( new InputSource( urlDoc.openStream() ),
Employees.class, urlSchemaXSD );
  } // End of read()


} // *** End of class EmployeesReader{}

// *** End Of EmployeesReader.java
```

EmployeesWriter.java

```
/*
================================================================================================
===

                        I t c W o r k s   C o d e   G e n e r a t o r

                                2001-2007 by i Technologies Corp

    Source File Name: EmployeesWriter.java

    Author:

    Date Generated:   08-19-2007

    Time Generated:   07:13:56
```

```
================================================================================
===
*/

package com.mycompany.dvo;

import java.net.URL;
import java.io.File;
import com.itc.util.ItcResourceStoreFactory;
import com.itc.xml.ItcBeanToXml;
import javax.xml.schema.util.XmlDeSerializer;


public class EmployeesWriter
{


  /**
   * Writer

   */
  public static String toString( Employees objToWrite ) throws Exception
  {

    URL urlSchemaXSD = ItcResourceStoreFactory.getInstance().getStore().getDocument(
"employees.xsd" );

    ItcBeanToXml btx = new ItcBeanToXml();

    btx.setFeature( XmlDeSerializer.ATTRIBUTE_MODEL, true  );

    btx.addSchema( urlSchemaXSD, Employees.class.getPackage() );
    btx.setFormattedOutput( true, 0 );
    btx.setContentMethods( Employee.class, "getAddress"  );
    btx.setContentMethods( Employees.class, "getEmployee"  );
    return btx.serialize( null, objToWrite  );
  } // End of toString()



  /**
   * Writer

   */
  public static void write( Employees objToWrite, File fileToWrite ) throws Exception
  {

    URL urlSchemaXSD = ItcResourceStoreFactory.getInstance().getStore().getDocument(
"employees.xsd" );

    ItcBeanToXml btx = new ItcBeanToXml();

    btx.setFeature( XmlDeSerializer.ATTRIBUTE_MODEL, true  );

    btx.addSchema( urlSchemaXSD, Employees.class.getPackage() );
    btx.setFormattedOutput( true, 0 );
    btx.setContentMethods( Employee.class, "getAddress"  );
    btx.setContentMethods( Employees.class, "getEmployee"  );
    btx.serialize( null, objToWrite, fileToWrite  );
  } // End of write()


} // *** End of class EmployeesWriter{}

// *** End Of EmployeesWriter.java
```

Additional ItcAttributes:

Two additional attributes: ***collection*** and ***choiceClassType*** have been added to the Itc implementation of the **<xsd:choice>** schema type to help facilitate code generation. The ***collection*** attribute refers to an unbounded element definition in the schema document and is useful when adding different derived types to a common list. The example below demonstrates this:

```
<xsd:schema>
  <xsd:complexType name="Employee">
    <xsd:sequence>
      <xsd:element name="contacts" type="tns:ContactType" maxOccurs="unbounded"/>
      <xsd:choice maxOccurs="unbounded" collection="contacts">
        <xsd:element name="email" type="tns:Email"/>
        <xsd:element name="homePhone" type="tns:Phone"/>
        <xsdLelement name="fax" type="tns:Fax"/>
      </xsd:choice>
   </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

The collection attribute (in bold above) refers to the ***contacts*** element, which defines a list of ContactType objects. ContactType is assumed to be a super class for the different types defined. The <xsd:choice> group element when defined with the ***collection*** attribute tells the code generator (ItcSchemaToJava described below) to generate an addXXXX method where XXXX is replaced by each element in the <xsd:choice> model group. The following code snippet is generated by the ItcSchemaToJava process:

```
public class Employee
{
  private List<ContactType>  m_listContacts;

  public void addEmail( Email email )
  {
    addToContactTypeList( email );
  }

  public void addHomePhone( Phone phone )
  {
    addToContactTypeList( phone );
  }

  private void addToContactList( ContactType contactType )
  {
    if ( m_listContacts == null )
        m_listContacts = new ArrayList<ContactType>();
    m_listContacts.add( contactType );

  }

}
```

The ***choiceClassType*** attribute should be used when you have only one instance of an object and you want the property to refer to that objects super type. Lets say we want to add a property to the Employee schema above that maintains the primary contact type.
We add the following new choice definition to our schema:

```
     <xsd:choice choiceClassType="tns:ContactType">
       <xsd:element name="email" type="tns:Email"/>
       <xsd:element name="homePhone" type="tns:Phone"/>
```

 Copyright © 2006, I Technologies Corp

```
      <xsdLelement name="fax" type="tns:Fax"/>
    </xsd:choice>

// new methods added to Employee.class
public class Employee
{
  private ContactType            m_contactType;

  /**
   *
   */
  public void setEmail( Email email )
  {
    m_contactType = email;
  } // End of setEmail()



  /**
   *
   */
  public Email getEmail()
  {
    if ( m_contactType instanceof Email  )
      return (Email)m_contactType;

    return null;

  } // End of getEmail()

  // this pattern follows for each choice element type defined

}
```

ItcJavaToSchema

The ItcJavaToSchema utility creates XML Schema documents from Java class files or Java instances. The utility will follow the object graph if present. If you specify a Java class as input and a property returns a Collection, ItcJavaToSchema cannot determine the type of object in the collection so it generates the element with the type attribute set to question marks ('?'). If an object instance is used, the first object in the collection is used to determine the type. The utility supports a command line interface but only a Java class may be specified. The command line options are described below:

-***c (required)*** The package and class name to generate the schema from
-f (required) this is the path name of the output schema to be generated
-t (required) this specifies the Target Namespace for the schema
-p (optional) this specifies the Target Namespace prefix and defaults to "tns" if omitted

The following example shows a command line example. It creates a schema called person.xsd in the myschemas folder. The target namespace is http://mycompany.com and uses the default prefix of tns

java com.itc.xml.schema.tools.ItcJavaToSchema –c com.mycompany.dvo.Person –f /myschemas/person.xsd –t http://mycomapny.com

ItcXmlToXmlSchema

The ItcXmlToXmlSchema creates an XML Schema document (.xsd) from an XML instance document.

The command line options are described below:

-f (required) The path of the input XML instance document to parse
-o (required) The path of the output schema document to be created
-t (optional) The target namespace of the newly created schema document

Example:

```
java com.itc.xml.schema.tools.ItcXmlToXmlSchema -f /docs/myxml.xml -o
/docs/myschema.xsd -t http://mycompany.com/docs
```

The above example parses the input document /docs/myxml.xml document and creates the XML schema document /docs/myschema.xsd. The schema's target namespace is http://mycompany.com/docs.