

XML HACKS™

100 Industrial-Strength Tips & Tools



O'REILLY®

Michael Fitzgerald

HACK
#74

Create an XML Schema Document from an Instance or DTD

There are several tools that can help you generate an XML Schema document from either an instance or a DTD. This hack shows you how to get the job done with little fuss.

This hack walks you through the process of creating an XML Schema document from an XML document or a DTD. The DTD2XS utility uses a DTD, but the XSD Inference, Trang, Relaxer, and xmlspy tools all rely on an instance. Each of the following sections walks you through the simple steps necessary to get the job done with any of the tools.

LuMriX.net's DTD2XS

LuMriX.net (<http://www.lumrix.net>) offers a Java tool for creating an XML Schema from a DTD. Their tool is called DTD2XS (<http://sepia0.informatik.med.uni-giessen.de/dtd2xs.php>). This tool comes with a command-line and a browser interface. We'll cover the command-line interface in this hack.

Download the tool and extract the archive. In the archive, you will find the files `dtd2xs.class`, `dtd2xsd.class`, `xurl.class`, and `complextypes.xml`. Copy these files to the working directory and type the following command (which assumes that the working directory is in the classpath):

```
java dtd2xsd time.dtd
```

The tool will generate the following output (Example 5-10).

Example 5-10. DTD2XS tool output

```
dtd2xs: dtdURI file:///C:/Hacks/examples/time.dtd
dtd2xs: resolveEntities true
dtd2xs: ignoreComments true
dtd2xs: commentLength 100
dtd2xs: commentLanguage null
dtd2xs: conceptHighlight 2
dtd2xs: conceptOccurrence 1
dtd2xs: conceptRelation element attribute
dtd2xs: load DTD ... done
dtd2xs: remove comments from DTD ... done
dtd2xs: DOM translation ...
... done
dtd2xs: complextypes.xml ... done
dtd2xs: add namespace ... done<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="time">
<xs:complexType>
<xs:sequence>
<xs:element ref="hour"/>
<xs:element ref="minute"/>
```

Example 5-10. DTD2XS tool output (continued)

```
<xs:element ref="second"/>
<xs:element ref="meridiem"/>
<xs:element ref="atomic"/>
</xs:sequence>
<xs:attribute name="timezone" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
<xs:element name="hour" type="xs:string"/>
<xs:element name="minute" type="xs:string"/>
<xs:element name="second" type="xs:string"/>
<xs:element name="meridiem" type="xs:string"/>
<xs:element name="atomic">
<xs:complexType>
<xs:attribute name="signal" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
</xs:schema>
```

You can redirect the XML Schema to a file using >:

```
java dtd2xsd time.dtd > newtime.xsd
```

Microsoft XSD Inference 1.0

Microsoft offers a tool that infers an XML Schema document from an instance. It's called the XSD Inference 1.0 tool. It can be run online (<http://apps.gotdotnet.com/xmltools/xsdinference/>) and is also available for download (<http://apps.gotdotnet.com/xmltools/xsdinference/XSDInference.exe>). In an effort to be brief, I'll demonstrate only the online version.

The online version is shown in Internet Explorer in [Figure 5-5](#). (For a usage overview, see <http://apps.gotdotnet.com/xmltools/xsdinference/overview.html>.) To infer a schema from an instance, click the Browse button on the form. Then search for and select *time.xml* from the "Choose file" dialog box. Click Open. Now click the Infer Schema button. (Click the Refine Schema button to refine a previously inferred schema. If you click Refine Schema without first clicking Infer Schema, the result will be the same as clicking Infer Schema.) The result is displayed in the "View the XML Schema(s) generated" text area ([Figure 5-6](#)). Any errors would be displayed in the "View the error log" text area.

Trang

You can download the current Trang JAR (*trang.jar*) from <http://www.thaiopensource.com/download/>, then place the JAR in the working directory. In this section, I will cover how to create an XML Schema document from an instance. (If you need help with Java, refer to "Run Java Programs that Process XML" [\[Hack #10\]](#).)

Create an XML Schema Document from an Instance or DTD

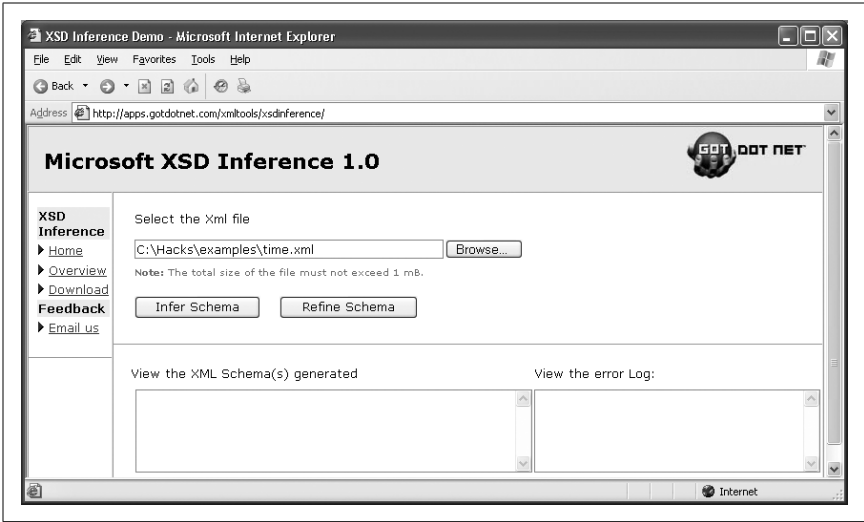


Figure 5-5. Microsoft XSD Inference tool in IE

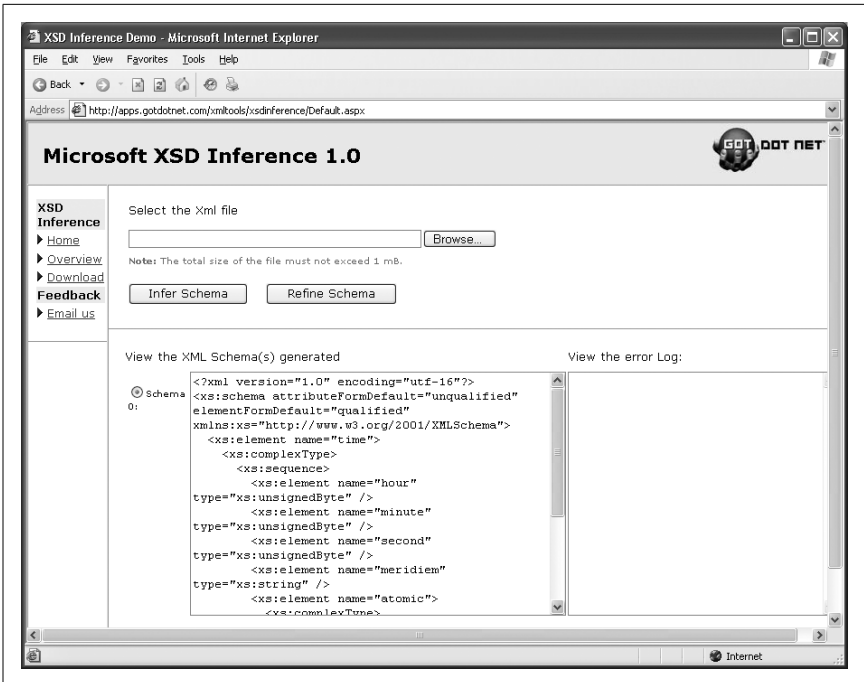


Figure 5-6. Results of inferring a schema with the XSD Inference tool in IE

To create an XML Schema from *time.xml*, use the following command:

```
java -jar trang.jar time.xml generated.xsd
```

In [Example 5-11](#), you can see what *generated.xsd* should look like.

Example 5-11. generated.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">
  <xs:element name="time">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="hour"/>
        <xs:element ref="minute"/>
        <xs:element ref="second"/>
        <xs:element ref="meridiem"/>
        <xs:element ref="atomic"/>
      </xs:sequence>
      <xs:attribute name="timezone" use="required" type="xs:NCName"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="hour" type="xs:integer"/>
  <xs:element name="minute" type="xs:integer"/>
  <xs:element name="second" type="xs:integer"/>
  <xs:element name="meridiem" type="xs:NCName"/>
  <xs:element name="atomic">
    <xs:complexType>
      <xs:attribute name="signal" use="required" type="xs:boolean"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Relaxer

Assuming that [Relaxer is already installed \[Hack #37\]](#), you can type the following command to generate an XML Schema document from *time.xml*:

```
relaxer -dir:out -xsd time.xml
```

Because Relaxer automatically uses the filename of the input file (*time.xml*) as the filename for the output file (*time.xsd*), we use the `-dir:out` option so that the output file will be stored in the *out* subdirectory. (If a subdirectory does not exist, Relaxer creates it.) The result of running Relaxer with `-xsd` is the file *out/time.xsd*, as shown in [Example 5-12](#).

Example 5-12. out/time.xsd

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns=""
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="">
  <xsd:element name="time" type="time"/>
  <xsd:complexType name="time">
    <xsd:sequence>
```

Example 5-12. *out/time.xsd* (continued)

```
<xsd:element name="hour" type="xsd:int"/>
<xsd:element name="minute" type="xsd:int"/>
<xsd:element name="second" type="xsd:int"/>
<xsd:element name="meridiem" type="xsd:token"/>
<xsd:element name="atomic" type="atomic"/>
</xsd:sequence>
<xsd:attribute name="timezone" type="xsd:token"/>
</xsd:complexType>
<xsd:complexType name="atomic">
  <xsd:sequence/>
  <xsd:attribute name="signal" type="xsd:boolean"/>
</xsd:complexType>
</xsd:schema>
```

Relaxer can use the content models of more than one XML document at a time, each with a different content model, in order to produce an XML Schema document. Use this command:

```
relaxer -dir:out -xsd time1.xml time.xml
```

When naming the output file, Relaxer uses the name of the first file in the list, so the output file will be *out/time1.xsd*, which is listed in [Example 5-13](#).

Example 5-13. *out/time1.xsd*

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns=""
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  targetNamespace="">
  <xsd:element name="time" type="time"/>
  <xsd:complexType name="time">
    <xsd:sequence>
      <xsd:element name="hour" type="xsd:int"/>
      <xsd:element name="minute" type="xsd:int"/>
      <xsd:element name="second" type="xsd:int"/>
      <xsd:element name="meridiem" type="xsd:token"/>
      <xsd:element maxOccurs="1" minOccurs="0" name="atomic"
        type="atomic"/>
    </xsd:sequence>
    <xsd:attribute name="timezone" type="xsd:token"/>
  </xsd:complexType>
  <xsd:complexType name="atomic">
    <xsd:sequence/>
    <xsd:attribute name="signal" type="xsd:boolean"/>
  </xsd:complexType>
</xsd:schema>
```

Notice the difference between the content models for the atomic element in these two examples. In [Example 5-13](#), the content for atomic is a `minOccurs` of 0 and a `maxOccurs` of 1. This is because atomic does not appear in *time1.xml*, so Relaxer interprets it as being optional rather than required.

xmlspy

As with a DTD, you can also generate an XML Schema document from an XML document with xmlspy (<http://www.xmlspy.com>). These instructions show you how to do this with xmlspy 2004 Enterprise Edition (Release 3).

1. Start xmlspy, and with File → Open, open *time.xml* from the working directory where you extracted the book's file archive.
2. Choose DTD/Schema → Generate DTD/Schema. The Generate DTD/Schema dialog box appears. Select the W3C Schema radio button and click OK ([Figure 5-7](#)).
3. You are then asked if you want to assign the generated DTD to the document. Click Yes. This adds an XML Schema instance namespace declaration and a `noNamespaceSchemaLocation` attribute to the XML document.
4. Then you are asked to save the schema. Save it as *spytime.xsd* and click the Save button. It is okay to overwrite *spytime.xsd* if it already exists in the working directory.
5. With File → Save As, save *time.xml*—now with a reference to the schema—as *spytimexsd.xml*.
6. Choose XML → Validate or press F8 to validate *spytimexsd.xml* against *spytime.xsd* ([Figure 5-8](#)).

You now have in hand a variety of tools to generate schemas for instances or DTDs. At least one of them should work for you!

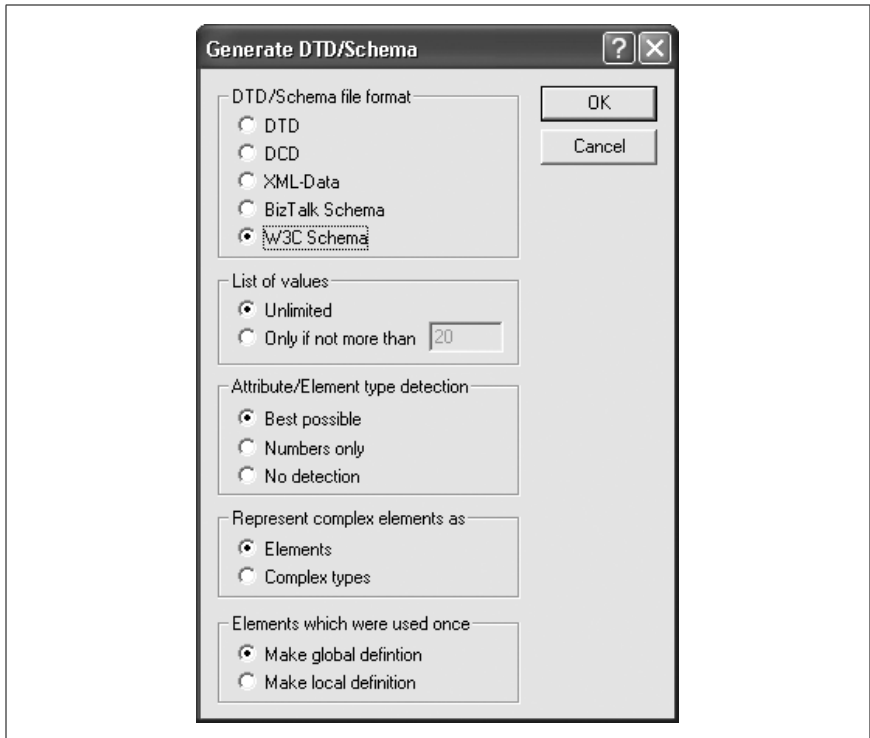


Figure 5-7. xmlspy's Generate DTD/Schema dialog box with W3C Schema selected

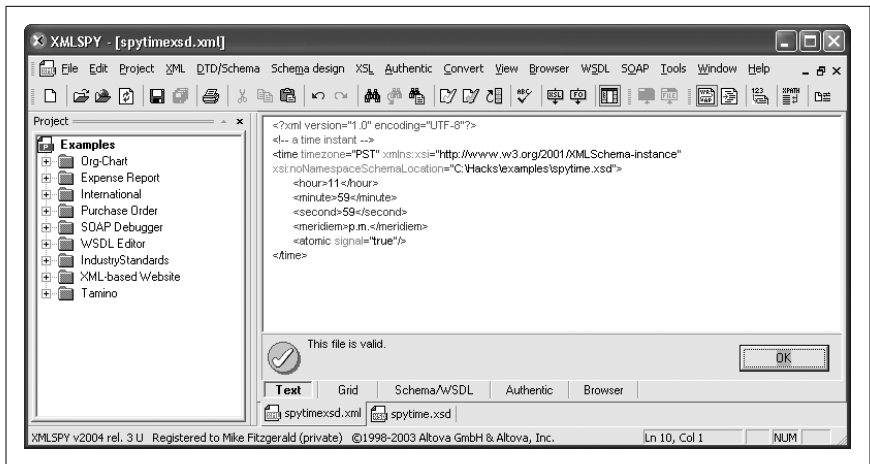


Figure 5-8. Validating spytimexsd.xml in xmlspy (text view)