XMLmind Xsdvalid Toolset
User's Guide
V3.0

www.xmlmind.com
xsdvalid-support@xmlmind.com
Pixware SARL
Immeuble Capricorne
23 rue Colbert
78180 Montigny Le Bretonneux
France
Phone: +33 (0)1 30 60 07 00
Fax: +33 (0)1 30 96 05 23

December 2, 2005

Contents

1 The xsdvalid toolset distribution **2**

2 Install **3**

2.1 Installing xsdvalid . . . 3

2.1.1 Requirements . . . 3

2.1.2 Install on Unix . . . 3

2.1.3 Install on Windows . . . 3

2.2 Content of the installation directory . . . 4

3 Xsdvalid command reference **5**

3.1 About the generated documentation . . . 6

4 Dtdvalid command reference **7**

5 Dtdtoxsd command reference **9**

6 Implementation limits **10**

6.1 Limitations related to XML Schema Datatypes . . . 10

6.2 Limitations related to XML Schema Structures . . . 10

6.3 Limitations related to DTD support . . . 12

7 Enhancements and bug fixes **14**

7.1 V3.0 Patch 1 (December 2, 2005) . . . 14

7.2 V2.11 (July 11, 2005) . . . 14

7.3 V2.10 (June 2, 2005) . . . 14

7.4 V2.9 Patch 1 (April 4, 2005) . . . 14

7.5 V2.9 (February 7, 2005) . . . 15

7.6 V2.6 Patch 1 (June 18, 2004) . . . 15

7.7 V2.6 (May 10, 2004) . . . 16

7.8 V2.5 Patch 3 (March 10, 2004) . . . 17

7.9 V2.5 Patch 1 (December 15, 2003) . . . 17

7.10 V2.5 (November 04, 2003) . . . 17

7.11 V2.4 (August 13, 2003) . . . 17

7.12 V2.0 Beta 1 (September 3, 2002) . . . 18

7.13 V1.0 Patch 3 (August 19, 2002) . . . 18

7.14 V1.0 Patch 2 (July 29, 2002) . . . 18

7.15 V1.0 Patch 1 (April 25, 2002) . . . 18

7.16 V1.0 (January 9, 2002) . . . 18

This guide describes three command-line tools for use by DTD and W3C XML Schema authors: xsdvalid, dtdvalid and dtdtoxsd. It is also a good reference for the support of DTD and W3C XML Schema in XXE.

1 The xsdvalid toolset distribution

This distribution contains the W3C XML Schema validation engine which is integrated in XMLmind XML Editor (XXE).

This engine has been made available to schema and DTD authors in the form of 3 command-line tools:

xsdvalid Checks an XML schema for validity. Checks an XML document for validity against an XML schema.

dtdvalid Checks a DTD for validity. Checks an XML document for validity against a DTD.

dtdtoxsd Converts a DTD to an XML schema.

Features:

- Fully implements W3C XML Schema *Validation Rules*.
- Implements the vast majority of *Schema Component Constraint* and *Schema Representation Constraint*. (Implementation limits are documented.)
- DTD and XML Schema support is provided by the same engine. In fact, the grammar part of a DTD (element, attribute and notation declarations) is loaded into the engine as if it was a schema.
- Once validated, a schema or a DTD can be serialized as a binary file for much faster reloading by XML document (instance) validation sessions.

Non features:

- Does not support anything related to *Schema Information Set Contribution*.
- This engine is designed to be an interactive, incremental validator and not a streaming one. Therefore, it needs to load the entire XML document into memory before attempting to perform validation.
- Xsdvalid, our W3C XML Schema validator, is a much better *instance validator* than a schema validator. As a schema validator, it is clearly too lenient and its error messages could be more detailed. However, it is still useful:
  - to quickly evaluate XMLmind XML Editor: if xsdvalid rejects your schema, you'd better pick another XML editor;
  - to get a second or third advice about the validity of your schema (given the complexity of the W3C recommendation, we recommend to check your schema using 2 or 3 different validators).

2 Install

2.1 Installing xsdvalid

2.1.1 Requirements

- Sun or Apple Java[tm] runtime 1.4.1 and above.
- At least 128Mb of memory and a 400MHz CPU.
- 10Mb of free disk space.

Xsdvalid tools have been tested with:

- Sun Java[tm] runtime 1.4.1+ (up to 1.5.0_05) under Windows NT4, 2000, XP Home, XP Professional (up to SP2), SuSE Linux 9.0 and 9.3.
- Mac OS X 10.3.9 and Java[tm] 1.4.2_09, 10.4.2 and Java[tm] 1.4.2_09.

2.1.2 Install on Unix

Procedure:

1. Make sure that the Java[tm] `bin/` directory is referenced in the $PATH and, at the same time, check that the Java[tm] runtime in the $PATH has the right version:

   ```
   $ java -version
   java version "1.5.0_04"
   Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_04-b05)
   Java HotSpot(TM) Client VM (build 1.5.0_04-b05, mixed mode, sharing)
   ```

2. Unpack the xsdvalid distribution somewhere.

   ```
   $ cd
   $ tar zxvf xsdvalid-30.tar.gz
   $ ls xsdvalid-30
   bin/
   doc/
   ```

3. Xsdvalid tools are intended to be used directly from the `xsdvalid-30/` directory. That is, you can add `xsdvalid-30/bin/` to your $PATH.

   ```
   $ xsdvalid-30/bin/xsdvalid -s my_w3c_xml_schema.xsd
   ```

2.1.3 Install on Windows

Manual install on Windows is similar to the install on Unix. Simply run `xsdvalid-30\bin\xsdvalid.bat`, `dtdvalid.bat`, `dtdtoxsd.bat` rather than the `xsdvalid-30/bin/xsdvalid`, `dtdvalid`, `dtdtoxsd` shell scripts.

2.2 Content of the installation directory

doc/xsdvalid Contains this user guide.

doc/xsdvalid/samples/xsdvalid Contains some files that can be used to quickly test xsdvalid. The examples given in the xsdvalid command reference section below make use of the files found in this directory.

doc/xsdvalid/samples/dtdvalid Contains some files that can be used to quickly test dtdvalid and dtdtoxsd. The examples given in the dtdvalid and dtdtoxsd command reference sections below make use of the files found in this directory.

bin/xsdvalid, dtdvalid, dtdtoxsd, xsdvalid.bat, dtdvalid.bat, dtdtoxsd.bat Scripts used to run xsdvalid, dtdvalid and dtdtoxsd. Use `xsdvalid`, `dtdvalid` and `dtdtoxsd` on any Unix system. Use `xsdvalid.bat`, `dtdvalid.bat` and `dtdtoxsd.bat` on Windows.

bin/*.jar All the (non-system) Java class libraries needed to run xsdvalid, dtdvalid and dtdtoxsd:

- xsdvalid.jar, contains the code of all the 3 command line tools.
- xp.jar, resolver.jar are needed to parse XML.

  These excellent packages have *not* been developed by XMLmind. Copyright information is contained in the corresponding `.LICENSE` file. Read the corresponding `.README` file to have more details about these packages.

3 Xsdvalid command reference

xsdvalid ?options? ?*xml_doc ... xml_doc*?

Checks an XML schema for validity. Checks an XML document for validity against an XML schema.

Options:

-o *file* Generate report in file *file*. Default: output report to console.

-s *schema* Use schema *schema* to validate XML documents. Default: use what is specified by xsi:schemaLocation and xsi:noNamespaceSchemaLocation.

-ss *namespace schema* Same as **-s** except that first parameter specifies the target namespace of the schema. This target namespace is *mandatory* if the schema is to be deserialized.

-r *dir* Load serialized schemas from directory *dir* if found there, otherwise load schemas from their XML sources.

-w *dir* Serialize loaded schemas to directory *dir*.

-gendoc *dir* Generate documentation in directory *dir*.

See note about the generated documentation.

-css *URL* Add link to specified CSS URL in generated documentation.

-v Be verbose. Default: be quiet.

It is possible to specify several **-s** and **-ss** options. Such multiple schemas (and their included/imported schemas, if any) are merged into one big global schema.

This command is XML catalog aware. This command will use the XML catalogs specified in environment variable `XML_CATALOG_FILES`. This variable must contain one or several XML catalog file names or URLs separated by a semi-colon (';').

XML catalogs may be used to resolve URLs found in the following places:

- Schema locations in `xsi:schemaLocation` and in `xsi:noNamespaceSchemaLocation`.
- Schema locations in `xs:include`, `xs:redefine`, `xs:import`.
- DTD locations in `<!DOCTYPE>` (typically used to define character entities).

Limitations:

- xsdvalid will not found xsi:schemaLocation and xsi:noNamespaceSchemaLocation attributes if specified on a element other that the root element of the XML instance.

Examples:

- Validate schema bugreport.xsd (which happens to import xhtml.xsd).

```
$ xsdvalid -s bugreport.xsd
```

- Validate XML document sample.xml against the schemas specified in the xsi:schemaLocation and xsi:noNamespaceSchemaLocation attributes.

```
$ xsdvalid sample.xml
```

- Validate XML document bad.xml against the schema bugreport.xsd, possibly overriding the schemas specified in the xsi:schemaLocation and xsi:noNamespaceSchemaLocation attributes.

```
$ xsdvalid -s bugreport.xsd  bad.xml
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:9:2:
element contains invalid data: "M1.2p" does not match pattern
"^((M|V)\d+\.\d+(p\d+)?)$"
[cvc-pattern-valid] [cvc-type.3.1.3]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:11:2:
element contains invalid data: syntax error in dateTime value
"2001-13-16T12:00:00"
[cvc-datatype-valid.1.2.1] [cvc-type.3.1.3]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:12:15:
the sequence of child elements is incorrect [cvc-complex-type.2.4]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:12:15:
element cannot contain element "html:font" [cvc-complex-type]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:24:0:
the sequence of child elements is incorrect [cvc-complex-type.2.4]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:36:0:
element has no attribute "number" [cvc-complex-type.3]
file:/home/hussein/src/xxe/distrib/samples/xsdvalid/bad.xml:E:37:2:
element contains invalid data: "xsd" is not one of the allowed values
[cvc-enumeration-valid] [cvc-type.3.1.3]
```

- Serialize bugreport.xsd (merged with imported schema xhtml.xsd) to directory serial.

```
$ xsdvalid -s bugreport.xsd  -w serial
$ ls serial/
directory.txt
schema0.ser
```

- Validate sample.xml against bugreport.xsd, deserialized from directory serial.

```
$ xsdvalid -v -r serial \
        -ss http://www.xmlmind.com/xmleditor/schema/bugreport \
        bugreport.xsd sample.xml
Deserializing schema 'http://www.xmlmind.com/xmleditor/schemas' (1582ms)
Loading XML document 'sample.xml' (457ms)
Validating 'sample.xml' (182ms)
```

  Note that sample.xml has an xsi:schemaLocation attribute, therefore there is no need to use option **-ss** even when the schema is to be deserialized.

```
$ xsdvalid -v -r serial sample.xml
Loading XML document 'sample.xml' (491ms)
Deserializing schema 'http://www.xmlmind.com/xmleditor/schemas' (1533ms)
Validating 'sample.xml' (189ms)
```

3.1 About the generated documentation

The generated HTML reference manual, organized like "*DocBook: The Definitive Guide*" by Norman Walsh and al., lists all elements and attributes specified in the W3C XML schema or DTD.

This manual is intended to help content authors create instances conforming to a given XML schema or DTD. This manual is not intended to help XML schema or DTD authors document their design.

Note that, for now, the documentation generator cannot extract documentation contained in a schema (i.e. in `annotation`/`documentation` elements) and merge extracted documentation with automatically generated documentation.

4 Dtdvalid command reference

dtdvalid ?options? ?*xml_doc ... xml_doc*?

Checks a DTD for validity. Checks an XML document for validity against a DTD.

Options:

-o *file* Generate report in file *file*. Default: output report to console.

-d *dtd* Use DTD *dtd* to validate XML documents. Default: use DTD specified in the XML document.

-dd *pubid dtd* Same as **-d** except that first parameter specifies the public ID of the DTD. This public ID is mandatory if the DTD is to be serialized or deserialized.

-r *dir* Load serialized DTDs from directory *dir* if found there, otherwise load DTDs from their XML sources.

-w *dir* Serialize loaded DTDs to directory *dir*.

-gendoc *dir* Generate documentation in directory *dir*.

See note about the generated documentation.

-css *URL* Add link to specified CSS URL in generated documentation.

-v Be verbose. Default: be quiet.

When the **-d** or **-dd** command-line options are used, the constraint that the root element of an XML instance and the document element of the DTD must match is not checked.

This command is XML catalog aware. This command will use the XML catalogs specified in environment variable `XML_CATALOG_FILES`. This variable must contain one or several XML catalog file names or URLs separated by a semi-colon (';').

Notes:

- An XML instance which references entities (such as) must have a <!DOCTYPE> containing the declarations of these entities, even when the DTD to be used is specified with the **-d** or **-dd** options.
- A DTD document (that is, a file with the .dtd extension) is serialized with a dummy document element name called "dummy".

  Therefore, if an XML document to be validated references such serialized DTD in its <!DOCTYPE>, dtdvalid will always complain that the root element in the XML instance does not match document element named "dummy".

  The method to get rid of this false alert is to always specify such serialized DTD using the **-dd** command-line option.

Examples:

- Validate DTD xhtml1-strict.dtd.

```
$ dtdvalid -d xhtml1-strict.dtd
```

- Validate XML document sample.xhtml using the DTD specified in <!DOCTYPE>.

```
$ dtdvalid sample.xhtml
```

- Validate XML document bad.xhtml against xhtml1-strict.dtd, possibly overriding the DTD specified in <!DOCTYPE>.

```
$ dtdvalid -d xhtml1-strict.dtd bad.xhtml
file:/home/hussein/src/xxe/distrib/samples/dtdvalid/bad.xhtml:E:7:2:
element contains characters other than white space [cvc-complex-type.2.3]
file:/home/hussein/src/xxe/distrib/samples/dtdvalid/bad.xhtml:E:8:4:
element has no attribute "align" [cvc-complex-type.3]
file:/home/hussein/src/xxe/distrib/samples/dtdvalid/bad.xhtml:E:13:4:
the sequence of child elements is incorrect [cvc-complex-type.2.4]
file:/home/hussein/src/xxe/distrib/samples/dtdvalid/bad.xhtml:E:13:4:
element cannot contain element "hr" [cvc-complex-type]
```

- Serialize DTD xhtml1-strict.dtd to directory serial.

```
$ dtdvalid -dd "-//W3C//DTD XHTML 1.0 Strict//EN" xhtml1-strict.dtd -w serial
$ ls serial/
directory.txt
schema0.ent
schema0.ser
```

- Validate XML document sample.xhtml using xhtml1-strict.dtd, deserialized from directory serial.

```
$ dtdvalid -v -r serial \
    -dd "-//W3C//DTD XHTML 1.0 Strict//EN" xhtml1-strict.dtd sample.xhtml
Deserializing global DTD '-//W3C//DTD XHTML 1.0 Strict//EN' (1079ms)
Loading XML document 'sample.xhtml' (748ms)
Validating 'sample.xhtml' (180ms)
```

5 Dtdtoxsd command reference

dtdtoxsd ?options? *in_dtd_file out_xsd_file*

Converts DTD *in_dtd_file* to XML-Schema *out_xsd_file*.

Options:

-t *namespace* Use *namespace* as the target namespace of the generated XML-Schema. Default: none (generated schema has no target namespace).

If the DTD declares text, external or unparsed entities, these declarations are copied to a file which has the same basename as *out_xsd_file* but with extension `.ent`. This file is created in the same directory as *out_xsd_file*.

In addition to *out_xsd_file*, a schema file named `xml.xsd` is created in the same directory as *out_xsd_file*. This secondary schema declares standard attributes xml:space, xml:lang and xml:base. The main schema always imports `xml.xsd` even if it doesn't reference any of the standard attributes.

This command is XML catalog aware. This command will use the XML catalogs specified in environment variable `XML_CATALOG_FILES`. This variable must contain one or several XML catalog file names or URLs separated by a semi-colon (`';'`).

Limitations:

- Parameter entities found in the DTD are not converted to group or attributeGroup XML-Schema components.
- A DTD declaring elements or attributes whose name contains the ":" character is silently converted to an invalid XML-Schema.
- Attributes whose name begins with (case-insensitive) string "xml", such as xmlns-style attributes, are silently skipped.

Examples:

- Convert the XHTML DTD to an XML Schema.

```
$ dtdtoxsd -t http://www.w3.org/1999/xhtml xhtml1-strict.dtd /tmp/xhtml.xsd
$ ls /tmp
xhtml.ent
xhtml.xsd
xml.xsd
```

6 Implementation limits

6.1 Limitations related to XML Schema Datatypes

Formal reference: XML Schema Part 2: Datatypes.

- A 32-bit signed integer is used rather than an arbitrary precision integer to implement the **length**, **minLength**, **maxLength**, **totalDigits** and **fractionDigits** facets.
- Similarly, the components of the **duration** datatype are implemented using 32-bit integers and double-precision floating-point numbers.
- The **length** facet of datatype **QName** is implemented as the number of characters in the local part of the name (that is, the prefix part is not taken into account by facet **length**).

6.2 Limitations related to XML Schema Structures

Formal reference: XML Schema Part 1: Structures.

Constraints on XML instances which are not checked:

- Entity Name: an attribute value of type **ENTITY** or **ENTITIES** must match the name of an unparsed entity declared in the DTD.

Constraints on XML schemas which are not checked:

- "The {model group} of the model group definition which corresponds to it per XML Representation of Model Group Definition Schema Components (§3.7.2) must be a ·valid restriction· of the {model group} of that model group definition in I, as defined in Particle Valid (Restriction) (§3.9.6)." [src-redefine.6.2.2]

  In this case, the implementation simply overwrites the previously defined **group**.
- "The {attribute uses} and {attribute wildcard} of the attribute group definition which corresponds to it per XML Representation of Attribute Group Definition Schema Components (§3.6.2) must be ·valid restrictions· of the {attribute uses} and {attribute wildcard} of that attribute group definition in I." [src-redefine.7.2.2]

  In this case, the implementation simply overwrites the previously defined **attributeGroup**.
- Attribute Group Definition Representation OK. [src-attribute_group.2] [src-attribute_group.3]
  Attribute Group Definition Properties Correct. [ag-props-correct.2] [ag-props-correct.3]

  attributeGroups are not validated as such. If something is wrong, it is detected when the the **attributeGroup** is actually used.

  Example 1: circular references are checked when the **attributeGroup** is actually used.

  Example 2: duplicate attribute and several ID attributes in the same **attributeGroup** are checked when the **attributeGroup** is actually used.
- Model Group Definition Representation OK. [mgd-props-correct]

  groups are not validated as such. If something is wrong, it is detected when the the **group** is actually used.
- Unique Particle Attribution. [cos-nonambig]
- Derivation Valid (Restriction, Simple). [cos-st-restricts.1.3] [cos-st-restricts.2.3.3] [cos-st-restricts.3.3.3]

  The implementation allows to add facets not defined by the base type.

- "It must in principle be possible to derive the complex type definition in two steps, the first an extension and the second a restriction (possibly vacuous), from that type definition among its ancestors whose {base type definition} is the ·ur-type definition·." [cos-ct-extends.1.5]
- "The {content type} of the {base type definition} must be a simple type definition of which the {content type} is a ·valid restriction· as defined in Derivation Valid (Restriction, Simple) (§3.14.6)." [derivation-ok-restriction.5.1.1]
  - See above the limitations related to Derivation Valid (Restriction, Simple) [cos-st-restricts]
  - In this case, the implementation does not check that the new facet value actually restricts the facet value of the base type.
- "No element member of the ·key-sequence· of any member of the ·qualified node set· was assessed as ·valid· by reference to an element declaration whose {nillable} is true." [cvc-identity-constraint.4.2.3]

Other specificities:

- The algorithm used to check *Particle Valid (Restriction)* [cos-particle-restrict] is more powerful than the one described in the spec.

  Rationale: the schema for schemas is found invalid when the algorithm described in the spec is used.
- `<xs:complexType name="title" mixed="true" />` is mixed, not empty.
- Not being able to load a schema **include**-ed, **import**-ed or **redefine**-ed by another schema is considered to be a fatal error [x-src-include.1] [x-src-import.1] [x-src-redefine.1]. For the spec, it is just a warning.
- A **import** construct must almost always specify a **schemaLocation** [x-src-import].

  However, the validation engine supports `xs:import` elements *without* a `schemaLocation` attribute, if an `xs:import` element for the same namespace but this time having a `schemaLocation` attribute has previously been processed.

  Example:

```
<xs:import namespace="foo"
           schemaLocation="http://foo.com/schema1.xsd" />

<!-- Later, typically inside an included module. -->
<xs:import namespace="foo" />
```

  Note that the other example below will not work because the validation engine cannot guess which of `schema1.xsd` or `schema2.xsd` contains the components to be imported.

```
<xs:import namespace="foo"
           schemaLocation="http://foo.com/schema1.xsd" />

<!-- Later, typically inside an included module. -->
<xs:import namespace="foo"
           schemaLocation="http://foo.com/schema2.xsd" />

<!-- Later, typically inside another included module. -->
<xs:import namespace="foo" />
```

- The implementation issues a warning for the following case: the element declaration corresponding to an element particle is abstract and there are no non-abstract substitutions for this declaration [x-p-props-correct].
- Identity-constraint definition identities must be unique within an XML Schema [x-c-props-correct].

- A regular expression such as "`[a-zA-Z0-9-]`" is not supported as is. It must be rewritten like this: "`[a-zA-Z0-9\-]`".
- For readability, whitespace may be used in `selector` and `field` XPath expressions. But whitespace is only supported around '`|`' and not around all tokens as mandated by the W3C recommendation.

  That is, it is possible to specify this:

```
<xs:key name="truck1" >
  <xs:selector xpath="." />
  <xs:field xpath="truck/@number | truck/@plate" />
</xs:key>
```

  But not this:

```
<xs:key name="truck1" >
  <xs:selector xpath="." />
  <xs:field xpath="truck / @number | truck / @plate" />
</xs:key>
```

- The following *valid* element declaration is not supported.

```
<xs:element name="foo">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="bar" />
      <xs:element name="bar" form="qualified" type="xs:decimal" /> <!--NOT SUPPORTED-->
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="bar" type="xs:decimal"  />
```

  An *implementation limit* error `x-cos-element-consistent` is reported in that case.

- The following *valid* element declaration is not supported.

```
<xs:element name="foo">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="bar" type="xs:token" />
      <xs:element name="bar" type="xs:token" nillable="true" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

  An *implementation limit* error `x-cos-element-consistent` is reported in that case.

6.3 Limitations related to DTD support

Formal reference: Extensible Markup Language (XML) 1.0 (Second Edition).

Constraints on XML instances which are not checked:

- Entity Name: an attribute value of type **ENTITY** or **ENTITIES** must match the name of an unparsed entity declared in the DTD.

Constraints on DTDs which are not checked:

- Notation Declared: in an unparsed entity, the name after **NDATA** must match the declared name of a notation.

- Standalone Document Declaration.
- Proper Declaration/PE Nesting.
- Proper Group/PE Nesting.
- Proper Conditional Section/PE Nesting.

7 Enhancements and bug fixes

7.1 V3.0 Patch 1 (December 2, 2005)

Bug fixes:

- A `xs:keyref` identity constraint could reference a `xs:key` identity constraint, but not to a `xs:unique` identity constraint.
- An `xs:all` group must be the topmost group of a content model. Xsdvalid failed to enforce this constraint when the `xs:all` group was contained in a `xs:group`.
- Xsdvalid allowed element names to contain certain invalid characters. For example: it allowed to declare: `<element name="`$\mu - computer$`"/>`.
- If *schema1* includes *schema2* and *schema2* imports *schema3*, *schema1* could reference components from *schema3* without necessarily having a `<xs:import namespace="target namespace of schema3"/>` element.
- A value like 12.34E56 was accepted as being a valid `xs:decimal`.

7.2 V2.11 (July 11, 2005)

Bug fixes:

- There was a bug in the documentation of the XMLmind Xsdvalid Toolset. The documentation should read: the `xsdvalid`, `dtdvalid`, `dtdtoxsd` command-line tools are XML catalog aware. These command line tools will use the XML catalogs specified in environment variable `XML_CATALOG_FILES`. This variable must contain one or several XML catalog file names or URLs separated by a semi-colon (';').

7.3 V2.10 (June 2, 2005)

Bug fixes:

- Fixed a `NullPointerException` that occurred for union types having a member type with a pattern facet or an enumeration facet and another member type of the same base type, but having no pattern facet or enumeration facet.
- When `xsdvalid` cannot cope with an `import` element without a `schemaLocation` attribute, it reports an `x-src-import` *warning*. Previously, it used to report a `x-src-import` fatal error.

7.4 V2.9 Patch 1 (April 4, 2005)

Bug fixes:

- The `blockDefault` and `finalDefault` attributes were applied not only to the direct descendants of a `schema` element but also to components obtained from imported schemas.

7.5 V2.9 (February 7, 2005)

Bug fixes:

- Error: `element declaration "bar" inconsistent [cos-element-consistent]` was reported for the *valid* example below:

```
<xs:element name="foo">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="bar" type="xs:token" />
      <xs:element name="bar" type="xs:token" nillable="true" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

  An error is still reported for the above valid schema, but the error is now an *implementation error*: `implementation limit: element declaration "bar" differs from previous element declarations "bar" [x-cos-element-consistent]`.

- The implementation of wildcards was out of date.

  For example, `##other` meant (to make it simple) "any namespace, including absent, different from `targetNamespace`".

  In fact, `##other` means "a namespace must be specified and this namespace must be different from `targetNamespace`".

Regressions:

- Xsdvalid: W3C XML Schemas having no target namespace can no longer be serialized. Option `-ss -` *schema_location* where '-' means no target namespace is no longer supported.

- Xsdvalid, dtdvalid: last deseralized schema or DTD is no longer cached. In practice, this means that:

  ```
  xsdvalid -r serial f1.xhtml f2.xhtml f3.xhtml
  ```

  is now much slower now because the W3C XML Schema for XHTML is deserialized for each document to be validated.

  If you want to speed up this, run something like:

  ```
  xsdvalid -r serial -ss http://www.xmlmind.com/xmleditor/schema/xhtml xhtml.xsd \
      f1.xhtml f2.xhtml f3.xhtml
  ```

7.6 V2.6 Patch 1 (June 18, 2004)

Enhancements:

- Upgraded the Schema for Schemas to the version included in "*XML Schema Part 1: Structures Second Edition - W3C Proposed Edited Recommendation 18 March 2004*".

  This version of the Schema for Schemas allows to add attributes with non-schema namespaces to annotate most schema components.

  Slightly edited this normative schema to allow spaces before and after path alternatives in the `xpath` attributes of elements `selector` and `field`. Example:

```
<xs:key name="truck1" >
  <xs:selector xpath="truck" />
  <xs:field xpath="@number | @plate" />
</xs:key>
```

- Xsdvalid now supports `xs:import` elements *without* a `schemaLocation` attribute, if an `xs:import` element for the same namespace but this time having a `schemaLocation` attribute has previously been processed.

  Example:

```
  <xs:import namespace="foo"
             schemaLocation="http://foo.com/schema1.xsd" />

  <!-- Later, typically inside an included module. -->
  <xs:import namespace="foo" />
```

  Note that the other example below will not work because the W3C XML Schema validation engine cannot guess which of `schema1.xsd` or `schema2.xsd` needs to be imported.

```
  <xs:import namespace="foo"
             schemaLocation="http://foo.com/schema1.xsd" />

  <!-- Later, typically inside an included module. -->
  <xs:import namespace="foo"
             schemaLocation="http://foo.com/schema2.xsd" />

  <!-- Later, typically inside another included module. -->
  <xs:import namespace="foo" />
```

- Xsdvalid: it is now possible to use XML catalogs to resolve URLs found in the following places:
  - Schema locations in `xsi:schemaLocation` and in `xsi:noNamespaceSchemaLocation`.
  - Schema locations in `xs:include`, `xs:redefine`, `xs:import`.

Bug fixes:

- "Placeholder groups" such as:

```
<xs:group name="Misc.extra">
  <xs:choice/>
</xs:group>
```

  caused xsdvalid to throw a `NullPointerException`.

- Identity constraints found in element declarations themselves contained in groups, like for example,

```
   <xs:group name="general">
      <xs:sequence>
         <xs:element name="general" type="general">
            <xs:unique name="generalUnique">
               <xs:selector xpath="*"/>
               <xs:field xpath="@uniqueElementName"/>
            </xs:unique>
         </xs:element>
      </xs:sequence>
   </xs:group>
```

  caused xsdvalid to report false errors (the error message was: an identity-constraint with the same name "*XXX*" has already been defined).

7.7 V2.6 (May 10, 2004)

Enhancements:

- The `xsdvalid`, `dtdvalid`, `dtdtoxsd` command-line tools are now XML catalog aware. These command line tools will use the XML catalogs specified in environment variable `XXE_CATALOG`. This variable must contain one or several XML catalog file names or URLs separated by a semi-colon (';').

Bug fixes:

- The complete schema built from *multiple* schemas specified using `xsi:schemaLocation` and / or `xsi:noNamespaceSchemaLocation` attributes was incorrect.

7.8 V2.5 Patch 3 (March 10, 2004)

Bug fixes:

- The validation engine implemented the original "*Schema Component Constraint: Type Derivation OK (Simple)*" specification, not the one fixed in "*XML Schema 1.0 Specification Errata*". This caused valid substitution groups where the type of the head was `xs:anyType` to be rejected by the engine.
- A substitution group containing just a single non-abstract member was ignored.

7.9 V2.5 Patch 1 (December 15, 2003)

Bug fixes:

- In XPath expressions, the default namespace declared with `xmlns` was used for name tests on elements. Now, if the QName does not have a prefix, then the namespace URI is null.

  Example: default namespace is "`http://www.w3.org/1999/xhtml`".

  Before the bug fix, "`div/p`" meant
  "`{http://www.w3.org/1999/xhtml}div/{http://www.w3.org/1999/xhtml}p`".

  After the bug fix, "`div/p`" means
  "`{}div/{}p`".

7.10 V2.5 (November 04, 2003)

Option **-gendoc** added to **xsdvalid** and **dtdvalid** allows to automatically generate an hypertext (HTML) reference manual from an XML schema or a DTD. See note about the generated documentation.

7.11 V2.4 (August 13, 2003)

Enhancements:

- Full support of regular expressions as specified in http://www.w3.org/TR/xmlschema-2/#regexs thanks to James Clark's excellent xsdregex library (XSD to Java Regular Expression Translator).

Bug fixes:

- QNames in included schema with no target namespace were not properly resolved when the including schema had a target namespace (feature known as "Chameleon Includes").

7.12 V2.0 Beta 1 (September 3, 2002)

Changed version number to V2 to use the same version number as XXE.

7.13 V1.0 Patch 3 (August 19, 2002)

Forgot to update the documentation for the release of Patch2.

7.14 V1.0 Patch 2 (July 29, 2002)

Validating really large XML schemas on Windows was not possible due to a stack overflow error. Increasing the stack size by editing xsdvalid.bat and adding -Xss1m to the Java command line had no effect.

Xsdvalid 1.0 Patch2 requires Java 1.4. It will not run with Java 1.3. *Do not upgrade if you cannot install Java 1.4 on your machine.*

7.15 V1.0 Patch 1 (April 25, 2002)

Fixed an obscure bug related to restrictions of the NMTOKENS, IDREFS, and ENTITIES simple types.

7.16 V1.0 (January 9, 2002)

Initial release.