Namespaces in XML

1.0 (Third Edition)

W3C Recommendation 8 December 2009

This version:

http://www.w3.org/TR/2009/REC-xml-names-20091208/

Latest version:

http://www.w3.org/TR/xml-names/

Previous versions:

http://www.w3.org/TR/2006/REC-xml-names-20060816/
http://www.w3.org/TR/2009/PER-xml-names-20090806/

Authors and Contributors:

Tim Bray (Textuality) <tbray@textuality.com>
Dave Hollander (Contivo, Inc.) <dmh@contivo.com>
Andrew Layman (Microsoft) <andrewl@microsoft.com>
Richard Tobin (University of Edinburgh and Markup Technology Ltd) <richard@inf.ed.ac.uk>
Henry S. Thompson (University of Edinburgh and W3C) <ht@w3.org>

Abstract

XML namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URI references.

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This document is a product of the XML Core Working Group as part of the W3C XML Activity. The English version of this specification is the only normative version. However, for translations of this document, see http://www.w3.org/2003/03/Translations/byTechnology?technology=xml-names .

Known implementations are documented in the Namespaces 1.1 implementation report (all known Namespaces 1.1 implementations also support Namespaces 1.0) . A test suite is also available via the XML Test Suite page.

This third edition incorporates all known errata as of the publication date. It supersedes the previous edition of 16 August 2006.

This edition has been widely reviewed. Only minor editorial changes have been made since the 6 August 2009 Proposed Edited Recommendation.

Please report errors in this document to xml-names-editor@w3.org; public archives are available. The errata list for this document is available at http://www.w3.org/XML/2009/xml-names-errata .

This document has been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and is endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

Appendices

This page is intentionally left blank.

1. Motivation and Summary

We envision applications of Extensible Markup Language (XML) where a single XML document may contain elements and attributes (here referred to as a "markup vocabulary") that are defined for and used by multiple software modules. One motivation for this is modularity: if such a markup vocabulary exists which is well-understood and for which there is useful software available, it is better to re-use this markup rather than re-invent it.

Such documents, containing multiple markup vocabularies, pose problems of recognition and collision. Software modules need to be able to recognize the elements and attributes which they are designed to process, even in the face of "collisions" occurring when markup intended for some other software package uses the same element name or attribute name.

These considerations require that document constructs should have names constructed so as to avoid clashes between names from different markup vocabularies. This specification describes a mechanism, *XML namespaces*, which accomplishes this by assigning expanded names to elements and attributes.

1.1. A Note on Notation and Usage

Where *EMPHASIZED*, the key words *MUST*, *MUST NOT*, *REQUIRED*, *SHOULD*, *SHOULD NOT*, *MAY* in this document are to be interpreted as described in [Keywords].

Note that many of the nonterminals in the productions in this specification are defined not here but in the XML specification [XML]. When nonterminals defined here have the same names as nonterminals defined in the XML specification, the productions here in all cases match a subset of the strings matched by the corresponding ones there.

In this document's productions, the `NSC` is a "Namespace Constraint", one of the rules that documents conforming to this specification *MUST* follow.

2. XML Namespaces

2.1. Basic Concepts

An *XML namespace* is identified by a URI reference [RFC3986]; element and attribute names may be placed in an XML namespace using the mechanisms described in this specification.

An *expanded name* is a pair consisting of a namespace name and a local name. For a name *N* in a namespace identified by a URI *I*, the *namespace name* is *I*. For a name *N* that is not in a namespace, the *namespace name* has no value. In either case the *local name* is *N*. It is this combination of the universally managed URI namespace with the vocabulary's local names that is effective in avoiding name clashes.

URI references can contain characters not allowed in names, and are often inconveniently long, so expanded names are not used directly to name elements and attributes in XML documents. Instead qualified names are used. A *qualified name* is a name subject to namespace interpretation. In documents conforming to this specification, element and attribute names appear as qualified names. Syntactically, they are either or . An attribute-based declaration syntax is provided to bind prefixes to namespace names and to bind a default namespace that applies to unprefixed element names; these declarations are scoped by the elements on which they appear so that different bindings may apply in different parts of a document. Processors conforming to this specification *MUST* recognize and act on these declarations and prefixes.

2.2. Use of URIs as Namespace Names

The empty string, though it is a legal URI reference, cannot be used as a namespace name.

The use of relative URI references, including same-document references, in namespace declarations is deprecated.

This deprecation of relative URI references was decided on by a W3C XML Plenary Ballot [Relative URI deprecation]. It also declares that "later specifications such as DOM, XPath, etc. will define no interpretation for them".

2.3. Comparing URI References

URI references identifying namespaces are compared when determining whether a name belongs to a given namespace, and whether two names belong to the same namespace. The two URIs are treated as strings, and they are *identical* if and only if the strings are identical, that is, if they are the same sequence of characters. The comparison is case-sensitive, and no %-escaping is done or undone.

A consequence of this is that URI references which are not identical in this sense may resolve to the same resource. Examples include URI references which differ only in case or %-escaping, or which are in external entities which have different base URIs (but note that relative URIs are deprecated as namespace names).

In a namespace declaration, the URI reference is the normalized value of the attribute, so replacement of XML character and entity references has already been done before any comparison.

Examples:

The URI references below are all different for the purposes of identifying namespaces, since they differ in case:

- `http://www.example.org/wine`
- `http://www.Example.org/wine`
- `http://www.example.org/Wine`

The URI references below are also all different for the purposes of identifying namespaces:

- `http://www.example.org/~wilbur`
- `http://www.example.org/%7ewilbur`
- `http://www.example.org/%7Ewilbur`

Because of the risk of confusion between URIs that would be equivalent if dereferenced, the use of %-escaped characters in namespace names is strongly discouraged.

3. Declaring Namespaces

A namespace (or more precisely, a namespace binding) is *declared* using a family of reserved attributes. Such an attribute's name must either be `xmlns` or begin `xmlns:`. These attributes, like any other XML attributes, may be provided directly or by default.

Attribute Names for Namespace Declaration

[1]	NSAttName	::=	**PrefixedAttName** | **DefaultAttName**	
[2]	PrefixedAttName	::=	'xmlns:' **NCName**	
[3]	DefaultAttName	::=	'xmlns'	
[4]	NCName	::=	Name - (Char* ':' Char*)	/* An XML Name, minus the ":" */

The attribute's normalized value *MUST* be either a URI reference — the namespace name identifying the namespace — or an empty string. The namespace name, to serve its intended purpose, *SHOULD* have the characteristics of uniqueness and persistence. It is not a goal that it be directly usable for retrieval of a schema (if any exists). Uniform Resource Names [RFC2141] is an example of a syntax that is designed with these goals in mind. However, it should be noted that ordinary URLs can be managed in such a way as to achieve these same goals.

If the attribute name matches **PrefixedAttName**, then the **NCName** gives the *namespace prefix*, used to associate element and attribute names with the namespace name in the attribute value in the scope of the element to which the declaration is attached.

If the attribute name matches **DefaultAttName**, then the namespace name in the attribute value is that of the *default namespace* in the scope of the element to which the declaration is attached. Default namespaces and overriding of declarations are discussed in § 6 – Applying Namespaces to Elements and Attributes on page 6.

An example namespace declaration, which associates the namespace prefix `edi` with the namespace name `http://ecommerce.example.org/schema`:

```
<x xmlns:edi='http://ecommerce.example.org/schema'>
  <!-- the "edi" prefix is bound to http://ecommerce.example.org/schema
       for the "x" element and contents -->
</x>
```

Namespace Constraint: Reserved Prefixes and Namespace Names

The prefix `xml` is by definition bound to the namespace name `http://www.w3.org/XML/1998/namespace`. It *MAY*, but need not, be declared, and *MUST NOT* be bound to any other namespace name. Other prefixes *MUST NOT* be bound to this namespace name, and it *MUST NOT* be declared as the default namespace.

The prefix `xmlns` is used only to declare namespace bindings and is by definition bound to the namespace name `http://www.w3.org/2000/xmlns/`. It *MUST NOT* be declared . Other prefixes *MUST NOT* be bound to this namespace name, and it *MUST NOT* be declared as the default namespace. Element names *MUST NOT* have the prefix `xmlns`.

All other prefixes beginning with the three-letter sequence x, m, l, in any case combination, are reserved. This means that:

- users *SHOULD NOT* use them except as defined by later specifications

- processors *MUST NOT* treat them as fatal errors.

Though they are not themselves reserved, it is inadvisable to use prefixed names whose LocalPart begins with the letters x, m, l, in any case combination, as these names would be reserved if used without a prefix.

4. Qualified Names

In XML documents conforming to this specification, some names (constructs corresponding to the nonterminal Name) *MUST* be given as qualified names, defined as follows:

Qualified Name

[5]	QName	::=	**PrefixedName** | **UnprefixedName**
[6]	PrefixedName	::=	**Prefix** ':' **LocalPart**
[7]	UnprefixedName	::=	**LocalPart**
[8]	Prefix	::=	**NCName**
[9]	LocalPart	::=	**NCName**

The **Prefix** provides the namespace prefix part of the qualified name, and *MUST* be associated with a namespace URI reference in a namespace declaration. The **LocalPart** provides the *local part* of the qualified name.

Note that the prefix functions *only* as a placeholder for a namespace name. Applications *SHOULD* use the namespace name, not the prefix, in constructing names whose scope extends beyond the containing document.

5. Using Qualified Names

In XML documents conforming to this specification, element names are given as qualified names, as follows:

Element Names

[10]	STag	::=	'<' **QName** (S **Attribute**)* S? '>'
[11]	ETag	::=	'</' **QName** S? '>'
[12]	EmptyElemTag	::=	'<' **QName** (S **Attribute**)* S? '/>'

An example of a qualified name serving as an element name:

```
  <!-- the 'price' element's namespace is http://ecommerce.example.org/schema -->
  <edi:price xmlns:edi='http://ecommerce.example.org/schema'
units='Euro'>32.18</edi:price>
```

Attributes are either namespace declarations or their names are given as qualified names:

Attribute

[13]	Attribute	::=	**NSAttName** Eq AttValue | **QName** Eq AttValue

An example of a qualified name serving as an attribute name:

```
<x xmlns:edi='http://ecommerce.example.org/schema'>
  <!-- the 'taxClass' attribute's namespace is http://ecommerce.example.org/schema
-->
  <lineItem edi:taxClass="exempt">Baby food</lineItem>
</x>
```

Namespace Constraint: Prefix Declared

The namespace prefix, unless it is `xml` or `xmlns`, *MUST* have been declared in a namespace declaration attribute in either the start-tag of the element where the prefix is used or in an ancestor element (i.e., an element in whose content the prefixed markup occurs).

Namespace Constraint: No Prefix Undeclaring

In a namespace declaration for a (i.e., where the is a), the attribute value *MUST NOT* be empty.

This constraint may lead to operational difficulties in the case where the namespace declaration attribute is provided, not directly in the XML document entity, but via a default attribute declared in an external entity. Such declarations may not be read by software which is based on a non-validating XML processor. Many XML applications, presumably including namespace-sensitive ones, fail to require validating processors. If correct operation with such applications is required, namespace declarations *MUST* be provided either directly or via default attributes declared in the internal subset of the DTD.

Element names and attribute names are also given as qualified names when they appear in declarations in the DTD:

Qualified Names in Declarations

[14]	doctypedecl	::=	'<!DOCTYPE' S **QName** (S ExternalID)? S? ('[' (markupdecl | PEReference | S)* ']' S?)? '>'
[15]	elementdecl	::=	'<!ELEMENT' S **QName** S contentspec S? '>'
[16]	cp	::=	(**QName** | choice | seq) ('?' | '*' | '+')?
[17]	Mixed	::=	'(' S? '#PCDATA' (S? '|' S? **QName**)* S? ')*' | '(' S? '#PCDATA' S? ')'
[18]	AttlistDecl	::=	'<!ATTLIST' S **QName AttDef*** S? '>'
[19]	AttDef	::=	S (**QName** | **NSAttName**) S AttType S DefaultDecl

Note that DTD-based validation is not namespace-aware in the following sense: a DTD constrains the elements and attributes that may appear in a document by their uninterpreted names, not by (namespace name, local name) pairs. To validate a document that uses namespaces against a DTD, the same prefixes must be used in the DTD as in the instance. A DTD may however indirectly constrain the namespaces used in a valid document by providing `#FIXED` values for attributes that declare namespaces.

6. Applying Namespaces to Elements and Attributes

6.1. Namespace Scoping

The scope of a namespace declaration declaring a prefix extends from the beginning of the start-tag in which it appears to the end of the corresponding end-tag, excluding the scope of any inner declarations with the same NSAttName part. In the case of an empty tag, the scope is the tag itself.

Such a namespace declaration applies to all element and attribute names within its scope whose prefix matches that specified in the declaration.

The expanded name corresponding to a prefixed element or attribute name has the URI to which the is bound as its namespace name, and the as its local name.

```
<?xml version="1.0"?>
<html:html xmlns:html='http://www.w3.org/1999/xhtml'>

  <html:head><html:title>Frobnostication</html:title></html:head>
  <html:body><html:p>Moved to
    <html:a href='http://frob.example.com'>here.</html:a></html:p></html:body>
</html:html>
```

Multiple namespace prefixes can be declared as attributes of a single element, as shown in this example:

```
<?xml version="1.0"?>
<!-- both namespace prefixes are available throughout -->
<bk:book xmlns:bk='urn:loc.gov:books'
         xmlns:isbn='urn:ISBN:0-395-36341-6'>
    <bk:title>Cheaper by the Dozen</bk:title>
    <isbn:number>1568491379</isbn:number>
</bk:book>
```

6.2. Namespace Defaulting

The scope of a default namespace declaration extends from the beginning of the start-tag in which it appears to the end of the corresponding end-tag, excluding the scope of any inner default namespace declarations. In the case of an empty tag, the scope is the tag itself.

A default namespace declaration applies to all unprefixed element names within its scope. Default namespace declarations do not apply directly to attribute names; the interpretation of unprefixed attributes is determined by the element on which they appear.

If there is a default namespace declaration in scope, the expanded name corresponding to an unprefixed element name has the URI of the default namespace as its namespace name. If there is no default namespace declaration in scope, the namespace name has no value. The namespace name for an unprefixed attribute name always has no value. In all cases, the local name is (which is of course the same as the unprefixed name itself).

```
<?xml version="1.0"?>
<!-- elements are in the HTML namespace, in this case by default -->
<html xmlns='http://www.w3.org/1999/xhtml'>
  <head><title>Frobnostication</title></head>
```

```
  <body><p>Moved to
    <a href='http://frob.example.com'>here</a>.</p></body>
</html>
```

```
<?xml version="1.0"?>
<!-- unprefixed element types are from "books" -->
<book xmlns='urn:loc.gov:books'
      xmlns:isbn='urn:ISBN:0-395-36341-6'>
    <title>Cheaper by the Dozen</title>
    <isbn:number>1568491379</isbn:number>
</book>
```

A larger example of namespace scoping:

```
<?xml version="1.0"?>
<!-- initially, the default namespace is "books" -->
<book xmlns='urn:loc.gov:books'
      xmlns:isbn='urn:ISBN:0-395-36341-6'>
    <title>Cheaper by the Dozen</title>
    <isbn:number>1568491379</isbn:number>
    <notes>
      <!-- make HTML the default namespace for some commentary -->
      <p xmlns='http://www.w3.org/1999/xhtml'>
          This is a <i>funny</i> book!
      </p>
    </notes>
</book>
```

The attribute value in a default namespace declaration *MAY* be empty. This has the same effect, within the scope of the declaration, of there being no default namespace.

```
<?xml version='1.0'?>
<Beers>
  <!-- the default namespace inside tables is that of HTML -->
  <table xmlns='http://www.w3.org/1999/xhtml'>
   <th><td>Name</td><td>Origin</td><td>Description</td></th>
   <tr>
     <!-- no default namespace inside table cells -->
     <td><brandName xmlns="">Huntsman</brandName></td>
     <td><origin xmlns="">Bath, UK</origin></td>
     <td>
       <details xmlns=""><class>Bitter</class><hop>Fuggles</hop>
         <pro>Wonderful hop, light alcohol, good summer beer</pro>
         <con>Fragile; excessive variance pub to pub</con>
         </details>
        </td>
      </tr>
```

```
    </table>
  </Beers>
```

6.3. Uniqueness of Attributes

Namespace Constraint: Attributes Unique

In XML documents conforming to this specification, no tag may contain two attributes which:

1. have identical names, or
2. have qualified names with the same local part and with prefixes which have been bound to namespace names that are identical.

This constraint is equivalent to requiring that no element have two attributes with the same expanded name.

For example, each of the `bad` empty-element tags is illegal in the following:

```
<!-- http://www.w3.org is bound to n1 and n2 -->
<x xmlns:n1="http://www.w3.org"
   xmlns:n2="http://www.w3.org" >
  <bad a="1"     a="2" />
  <bad n1:a="1"  n2:a="2" />
</x>
```

However, each of the following is legal, the second because the default namespace does not apply to attribute names:

```
<!-- http://www.w3.org is bound to n1 and is the default -->
<x xmlns:n1="http://www.w3.org"
   xmlns="http://www.w3.org" >
  <good a="1"     b="2" />
  <good a="1"     n1:a="2" />
</x>
```

7. Conformance of Documents

This specification applies to XML 1.0 documents. To conform to this specification, a document *MUST* be well-formed according to the XML 1.0 specification [XML].

In XML documents which conform to this specification, element and attribute names *MUST* match the production for **QName** and *MUST* satisfy the "Namespace Constraints". All other tokens in the document which are *REQUIRED*, for XML 1.0 well-formedness, to match the XML production for Name *MUST* match this specification's production for **NCName**.

A document is *namespace-well-formed* if it conforms to this specification.

It follows that in a namespace-well-formed document:

- All element and attribute names contain either zero or one colon;
- No entity names, processing instruction targets, or notation names contain any colons.

In addition, a namespace-well-formed document may also be namespace-valid.

A namespace-well-formed document is *namespace-valid* if it is valid according to the XML 1.0 specification, and all tokens other than element and attribute names which are *REQUIRED*, for XML 1.0 validity, to match the XML production for Name match this specification's production for **NCName**.

It follows that in a namespace-valid document:

- No attributes with a declared type of `ID`, `IDREF(S)`, `ENTITY(IES)`, or `NOTATION` contain any colons.

8. Conformance of Processors

To conform to this specification, a processor *MUST* report violations of namespace well-formedness, with the exception that it is not *REQUIRED* to check that namespace names are URI references [RFC3986].

A validating XML processor that conforms to this specification is *namespace-validating* if in addition it reports violations of namespace validity.

Appendix A. Normative References

Keywords

RFC 2119: Key words for use in RFCs to Indicate Requirement Levels, S. Bradner, ed. IETF (Internet Engineering Task Force), March 1997. Available at http://www.rfc-editor.org/rfc/rfc2119.txt

RFC2141

RFC 2141: URN Syntax, R. Moats, ed. IETF (Internet Engineering Task Force), May 1997. Available at http://www.rfc-editor.org/rfc/rfc2141.txt.

RFC3986

RFC 3986: Uniform Resource Identifier (URI): Generic Syntax, T. Berners-Lee, R. Fielding, and L. Masinter, eds. IETF (Internet Engineering Task Force), January 2005. Available at http://www.rfc-editor.org/rfc/rfc3986.txt

RFC3629

RFC 3629: UTF-8, a transformation format of ISO 10646, F. Yergeau, ed. IETF (Internet Engineering Task Force), November 2003. Available at http://www.rfc-editor.org/rfc/rfc3629.txt

XML

Extensible Markup Language (XML) 1.0, Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau eds. W3C (World Wide Web Consortium). Available at http://www.w3.org/TR/REC-xml/.

Appendix B. Other references (Non-Normative)

1.0 Errata

Namespaces in XML Errata. W3C (World Wide Web Consortium). Available at http://www.w3.org/XML/xml-names-19990114-errata.

1.0 2e Errata

Namespaces in XML (Second Edition) Errata. W3C (World Wide Web Consortium). Available at http://www.w3.org/XML/2006/xml-names-errata.

Relative URI deprecation

Results of W3C XML Plenary Ballot on relative URI References In namespace declarations 3-17 July 2000, Dave Hollander and C. M. Sperberg-McQueen, 6 September 2000. Available at http://www.w3.org/2000/09/xppa.

Appendix C. The Internal Structure of XML Namespaces (Non-Normative)

This appendix has been deleted.

Appendix D. Changes since version 1.0 (Non-Normative)

This version incorporates the errata as of 20 July 2009 [1.0 Errata] [1.0 2e Errata].

There are several editorial changes, including a number of terminology changes and additions intended to produce greater consistency. The non-normative appendix "The Internal Structure of XML Namespaces" has been removed. The BNF has been adjusted to interconnect properly with all editions of XML 1.0, including the fifth edition.

Appendix E. Acknowledgements (Non-Normative)

This work reflects input from a very large number of people, including especially the participants in the World Wide Web Consortium XML Working Group and Special Interest Group and the participants in the W3C Metadata Activity. The contributions of Charles Frankston of Microsoft were particularly valuable.

Appendix F. Orphaned Productions (Non-Normative)

The following two productions are modified versions of ones which were present in the first two editions of this specification. They are no longer used, but are retained here to satisfy cross-references to undated versions of this specification.

Because the `Letter` production of XML 1.0, originally used in the definition of `NCNameStartChar`, is no longer the correct basis for defining names since XML 1.0 Fifth Edition, the `NCNameStartChar`

production has been modified to give the correct results against any edition of XML, by defining `NCNameStartChar` in terms of **NCName**.

[20] NCNameChar ::= NameChar - ':' /* An XML NameChar, minus the ":" */

[21] NCNameStartChar ::= **NCName** - (Char Char Char*) /* The first letter of an **NCName** */

Production **NC-NCNameStartChar** takes advantage of the fact that a single-character NCName is necessarily an NCNameStartChar, and works by subtracting from the set of NCNames of all lengths the set of all strings of two or more characters, leaving only the NCNames which are one character long.

This page is intentionally left blank.

XML Schema: Understanding Namespaces by Rahul Srivastava. Moving to XML Schema? This introduction to namespaces will help you understand one of its more important components.Â As defined by the W3C Namespaces in XML Recommendation , an XML namespace is a collection of XML elements and attributes identified by an Internationalized Resource Identifier (IRI); this collection is often referred to as an XML "vocabulary." XML namespaces are defined by a W3C recommendation, dating 16 August 2006, called Namespaces in XML. XML tag names should be globally unique, as well as short for performance reasons. In order to resolve this conflict, the W3C namespace recommendation defines an attribute xmlns which can amend any XML element. If it is present in an element, it identifies the namespace for this element. The xmlns attribute has the following syntax: xmlns:prefix=namespace.