

XML Schema for WSML identifiers

Table of Contents

- [Schema Document Properties](#)
- [Global Schema Components](#)
 - [Simple Type: wsmIID](#)
 - [Simple Type: wsmIIDorLiteral-basic](#)
 - [Complex Type: wsmLiteral](#)
 - [Complex Type: wsmIIDorLiteral](#)
 - [Complex Type: wsmAnyID](#)
 - [Simple Type: wsmVariable](#)
 - [Simple Type: compositionIdentifier](#)
 - [Simple Type: arithmeticIdentifier](#)

[top](#)

Schema Document Properties

Target Namespace	http://www.wsmo.org/2004/wsmI
Element and Attribute Namespaces	<ul style="list-style-type: none">• Global element and attribute declarations belong to this schema's target namespace.• By default, local element declarations belong to this schema's target namespace.• By default, local attribute declarations have no namespace.
Documentation	version: \$Revision: 1.4 \$ date: \$Date: 2004/11/24 13:48:21 \$ author: Jos de Bruijn this schema is a module, which belongs to the WSML/XML schema specification. This schema provides the necessary definitions for the identifiers in WSML/XML.

Declared Namespaces

Prefix	Namespace
Default namespace	http://www.wsmo.org/2004/wsmI
xml	http://www.w3.org/XML/1998/namespace
xs	http://www.w3.org/2001/XMLSchema

Schema Component Representation

```
<xs:schema targetNamespace="http://www.wsmo.org/2004/wsmI "  
elementFormDefault="qualified" attributeFormDefault="unqualified">  
  ...  
</xs:schema>
```

[top](#)

Global Schema Components

Simple Type: **wsmIID**

<i>Super-types:</i>	None
<i>Sub-types:</i>	None

Name	wsmIID
Content	<ul style="list-style-type: none"> • Union of following types: <ul style="list-style-type: none"> ◦ xs:anyURI
Documentation	The basic kind of identifier in WSML: an IRI. Note that when translating the standard WSML syntax to XML, Qualified names are resolved to full IRIs.

Schema Component Representation

```
<xs:simpleType name="wsmIID">
  <xs:union memberTypes=" xs:anyURI" />
</xs:simpleType>
```

[top](#)

Simple Type: **wsmIIDorLiteral-basic**

<i>Super-types:</i>	None
<i>Sub-types:</i>	<ul style="list-style-type: none"> • wsmIIDorLiteral (by extension) • wsmIAnyID (by extension)

Name	wsmIIDorLiteral-basic
Content	<ul style="list-style-type: none"> • Union of following types: <ul style="list-style-type: none"> ◦ wsmIID ◦ xs:string
Documentation	A helper type for wsmIIDorLiteral; is not used directly

Schema Component Representation

```
<xs:simpleType name="wsmIIDorLiteral-basic">
  <xs:union memberTypes=" wsmIID xs:string" />
</xs:simpleType>
```

[top](#)

Complex Type: **wsmILiteral**

<i>Super-types:</i>	xs:string < wsmILiteral (by extension)
<i>Sub-types:</i>	None

Name	wsmLiteral
Abstract	no
Documentation	A literal in WSM. A literal can be typed or untyped. The recommended types are the XSD datatypes (http://www.w3.org/TR/xmlschema-2/) and <code>rdf:XMLLiteral</code> .

XML Instance Representation

```
<...
  type=" wsmID [0..1]">
    xs:string
</...>
```

Schema Component Representation

```
<xs:complexType name="wsmLiteral">
  <xs:simpleContent>
    <xs:extension base=" xs:string ">
      <xs:attribute name="type" type=" wsmID "/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

[top](#)

Complex Type: **wsmIDorLiteral**

<i>Super-types:</i>	wsmIDorLiteral-basic (by restriction) < wsmIDorLiteral (by extension)
<i>Sub-types:</i>	None

Name	wsmIDorLiteral
Abstract	no

XML Instance Representation

```
<...
  type=" wsmID [0..1]"
  kind=" xs:string (value comes from list: {'wsmID'|'literal'})
  [0..1]">
    wsmIDorLiteral-basic
</...>
```

Schema Component Representation

```
<xs:complexType name="wsmIDorLiteral">
  <xs:simpleContent>
    <xs:extension base=" wsmIDorLiteral-basic ">
      <xs:attribute name="type" type=" wsmID " use="optional"/>
      <xs:attribute name="kind" use="optional">
        <xs:simpleType>
          <xs:restriction base=" xs:string ">
            <xs:enumeration value="wsmID"/>
            <xs:enumeration value="literal"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

```
</xs:extension>
</xs:simpleContent>
</xs:complexType>
```

[top](#)

Complex Type: **wsmlAnyID**

Super-types: [wsmlIDorLiteral-basic](#) (by restriction) < **wsmlAnyID** (by extension)

Sub-types: None

Name	wsmlAnyID
Abstract	no
Documentation	wsmlAnyID corresponds with either an IRI, a variable, a literal or an anonymous identifier.

XML Instance Representation

```
<...
  type=" wsmlID [0..1]"
  kind=" xs:string (value comes from list:
  {'variable'|'wsmlID'|'literal'|'anonymousID'}) [1]">
  wsmlIDorLiteral-basic
</...>
```

Schema Component Representation

```
<xs:complexType name="wsmlAnyID">
  <xs:simpleContent>
    <xs:extension base=" wsmlIDorLiteral-basic ">
      <xs:attribute name="type" type=" wsmlID " use="optional"/>
      <xs:attribute name="kind" use="required">
        <xs:simpleType>
          <xs:restriction base=" xs:string ">
            <xs:enumeration value="variable"/>
            <xs:enumeration value="wsmlID"/>
            <xs:enumeration value="literal"/>
            <xs:enumeration value="anonymousID"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

[top](#)

Simple Type: **wsmlVariable**

Super-types: [xs:string](#) < **wsmlVariable** (by restriction)

Sub-types: None

Name	wsmlVariable
-------------	--------------

Content	<ul style="list-style-type: none"> • Base XSD Type: string • <i>pattern</i> = <code>'?'([a-z][A-Z][0-9][0x4E00 - 0x9FA5][0x3007][0x3021 - 0x3029])+</code>
Documentation	A variables in a WSMML logical expression.

Schema Component Representation

```
<xs:simpleType name="wsmmlVariable">
  <xs:restriction base="xs:string">
    <xs:pattern value="'?'([a-z][A-Z][0-9][0x4E00 - 0x9FA5][0x3007][0x3021 - 0x3029])+'"/>
  </xs:restriction>
</xs:simpleType>
```

[top](#)

Simple Type: **compositionIdentifier**

<i>Super-types:</i>	xs:string < compositionIdentifier (by restriction)
<i>Sub-types:</i>	None

Name	compositionIdentifier
Content	<ul style="list-style-type: none"> • Base XSD Type: string • <i>value</i> comes from list: {'eq' 'uneq' 'lt' 'lte' 'gt' 'gte'}
Documentation	The identifiers of in-built composition atoms.

Schema Component Representation

```
<xs:simpleType name="compositionIdentifier">
  <xs:restriction base="xs:string">
    <xs:enumeration value="eq"/>
    <xs:enumeration value="uneq"/>
    <xs:enumeration value="lt"/>
    <xs:enumeration value="lte"/>
    <xs:enumeration value="gt"/>
    <xs:enumeration value="gte"/>
  </xs:restriction>
</xs:simpleType>
```

[top](#)

Simple Type: **arithmaticIdentifier**

<i>Super-types:</i>	xs:string < arithmaticIdentifier (by restriction)
<i>Sub-types:</i>	None

Name	arithmaticIdentifier
-------------	----------------------

Content	<ul style="list-style-type: none">• Base XSD Type: string• <i>value</i> comes from list: {'add' 'sub' 'mul' 'div'}
Documentation	The identifiers of in-build arithmetic function symbols.

Schema Component Representation

```
<xs:simpleType name="arithmetIdentifier">  
  <xs:restriction base="xs:string">  
    <xs:enumeration value="add"/>  
    <xs:enumeration value="sub"/>  
    <xs:enumeration value="mul"/>  
    <xs:enumeration value="div"/>  
  </xs:restriction>  
</xs:simpleType>
```

[top](#)

Generated by [xs3p](#).