Built-in XSD Data Types

(source: http://w3schools.com)

XSD String Data Types

String data types are used for values that contains character strings.

String Data Type
The string data type can contain characters, line feeds, carriage returns, and tab characters.
The following is an example of a string declaration in a schema:

```
<xs:element name="customer" type="xs:string"/>
```

An element in your document might look like this:

```
<customer>John Smith</customer>
```

Or it might look like this:

```
<customer>      John Smith     </customer>
```

Note: The XML processor will not modify the value if you use the string data type.

NormalizedString Data Type
The normalizedString data type is derived from the String data type.
The normalizedString data type also contains characters, but the XML processor will remove line feeds, carriage returns, and tab characters.
The following is an example of a normalizedString declaration in a schema:

```
<xs:element name="customer" type="xs:normalizedString"/>
```

An element in your document might look like this:

```
<customer>John Smith</customer>
```

Or it might look like this:

```
<customer>     John Smith     </customer>
```

Note: In the example above the XML processor will replace the tabs with spaces.

Token Data Type
The token data type is also derived from the String data type.
The token data type also contains characters, but the XML processor will remove line feeds, carriage returns, tabs, leading and trailing spaces, and multiple spaces.
The following is an example of a token declaration in a schema:

```
<xs:element name="customer" type="xs:token"/>
```

An element in your document might look like this:

```
<customer>John Smith</customer>
```

Or it might look like this:

```
<customer>     John Smith     </customer>
```

Note: In the example above the XML processor will remove the tabs.

String Data Types
Note that all of the data types below derive from the String data type (except for string itself)!

Name	Description
ENTITIES	
ENTITY	
ID	A string that represents the ID attribute in XML (only used with schema attributes)
IDREF	A string that represents the IDREF attribute in XML (only used with schema attributes)
IDREFS	
language	A string that contains a valid language id
Name	A string that contains a valid XML name
NCName	
NMTOKEN	A string that represents the NMTOKEN attribute in XML (only used with schema attributes)
NMTOKENS	
normalizedString	A string that does not contain line feeds, carriage returns, or tabs
QName	
string	A string
token	A string that does not contain line feeds, carriage returns, tabs, leading or trailing spaces, or multiple spaces

Restrictions on String Data Types
Restrictions that can be used with String data types:

- enumeration
- length
- maxLength
- minLength
- pattern (NMTOKENS, IDREFS, and ENTITIES cannot use this constraint)
- whiteSpace

XSD Date and Time Data Types

Date and time data types are used for values that contain date and time.

Date Data Type
The date data type is used to specify a date.
The date is specified in the following form "YYYY-MM-DD" where:

- YYYY indicates the year
- MM indicates the month
- DD indicates the day

Note: All components are required!
The following is an example of a date declaration in a schema:

```
<xs:element name="start" type="xs:date"/>
```

An element in your document might look like this:

```
<start>2002-09-24</start>
```

Time Zones
To specify a time zone, you can either enter a date in UTC time by adding a "Z" behind the date - like this:

```
<start>2002-09-24Z</start>
```

or you can specify an offset from the UTC time by adding a positive or negative time behind the date - like this:

```
<start>2002-09-24-06:00</start>
```

or

```
<start>2002-09-24+06:00</start>
```

Time Data Type
The time data type is used to specify a time.
The time is specified in the following form "hh:mm:ss" where:

- hh indicates the hour
- mm indicates the minute
- ss indicates the second

Note: All components are required!
The following is an example of a time declaration in a schema:

```
<xs:element name="start" type="xs:time"/>
```

An element in your document might look like this:

```
<start>09:00:00</start>
```

Or it might look like this:

```
<start>09:30:10.5</start>
```

Time Zones
To specify a time zone, you can either enter a time in UTC time by adding a "Z" behind the time - like this:

```
<start>09:30:10Z</start>
```

or you can specify an offset from the UTC time by adding a positive or negative time behind the time - like this:

```
<start>09:30:10-06:00</start>
```

or

```
<start>09:30:10+06:00</start>
```

DateTime Data Type

The dateTime data type is used to specify a date and a time.
The dateTime is specified in the following form "YYYY-MM-DDThh:mm:ss" where:

- YYYY indicates the year
- MM indicates the month
- DD indicates the day
- T indicates the start of the required time section
- hh indicates the hour
- mm indicates the minute
- ss indicates the second

Note: All components are required!
The following is an example of a dateTime declaration in a schema:

```
<xs:element name="startdate" type="xs:dateTime"/>
```

An element in your document might look like this:

```
<startdate>2002-05-30T09:00:00</startdate>
```

Or it might look like this:

```
<startdate>2002-05-30T09:30:10.5</startdate>
```

Time Zones
To specify a time zone, you can either enter a dateTime in UTC time by adding a "Z" behind the time - like this:

```
<startdate>2002-05-30T09:30:10Z</startdate>
```

or you can specify an offset from the UTC time by adding a positive or negative time behind the time - like this:

```
<startdate>2002-05-30T09:30:10-06:00</startdate>
```

or

```
<startdate>2002-05-30T09:30:10+06:00</startdate>
```

Duration Data Type
The duration data type is used to specify a time interval.
The time interval is specified in the following form "PnYnMnDTnHnMnS" where:

- P indicates the period (required)
- nY indicates the number of years
- nM indicates the number of months
- nD indicates the number of days
- T indicates the start of a time section (required if you are going to specify hours, minutes, or seconds)
- nH indicates the number of hours
- nM indicates the number of minutes
- nS indicates the number of seconds

The following is an example of a duration declaration in a schema:

```
<xs:element name="period" type="xs:duration"/>
```

An element in your document might look like this:

```
<period>P5Y</period>
```

The example above indicates a period of five years.
Or it might look like this:

```
<period>P5Y2M10D</period>
```

The example above indicates a period of five years, two months, and 10 days.
Or it might look like this:

```
<period>P5Y2M10DT15H</period>
```

The example above indicates a period of five years, two months, 10 days, and 15 hours.
Or it might look like this:

```
<period>PT15H</period>
```

The example above indicates a period of 15 hours.

Negative Duration

To specify a negative duration, enter a minus sign before the P:

```
<period>-P10D</period>
```

The example above indicates a period of minus 10 days.

Date and Time Data Types

Name	Description
date	Defines a date value
dateTime	Defines a date and time value
duration	Defines a time interval
gDay	Defines a part of a date - the day (DD)
gMonth	Defines a part of a date - the month (MM)
gMonthDay	Defines a part of a date - the month and day (MM-DD)
gYear	Defines a part of a date - the year (YYYY)
gYearMonth	Defines a part of a date - the year and month (YYYY-MM)
time	Defines a time value

Restrictions on Date Data Types

Restrictions that can be used with Date data types:

- enumeration
- maxExclusive
- maxInclusive
- minExclusive
- minInclusive
- pattern
- whiteSpace

XSD Numeric Data Types

Decimal data types are used for numeric values.

Decimal Data Type
The decimal data type is used to specify a numeric value.
The following is an example of a decimal declaration in a schema:

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

An element in your document might look like this:

```
<prize>999.50</prize>
```

Or it might look like this:

```
<prize>+999.5450</prize>
```

Or it might look like this:

```
<prize>-999.5230</prize>
```

Or it might look like this:

```
<prize>0</prize>
```

Or it might look like this:

```
<prize>14</prize>
```

Note: The maximum number of decimal digits you can specify is 18.

Integer Data Type
The integer data type is used to specify a numeric value without a fractional component.
The following is an example of an integer declaration in a schema:

```
<xs:element name="prize" type="xs:integer"/>
```

An element in your document might look like this:

```
<prize>999</prize>
```

Or it might look like this:

```
<prize>+999</prize>
```

Or it might look like this:

```
<prize>-999</prize>
```

Or it might look like this:

```
<prize>0</prize>
```

Numeric Data Types
Note that all of the data types below derive from the Decimal data type (except for decimal itself)!

Name	Description
byte	A signed 8-bit integer
decimal	A decimal value
int	A signed 32-bit integer
integer	An integer value
long	A signed 64-bit integer
negativeInteger	An integer containing only negative values (..,-2,-1)
nonNegativeInteger	An integer containing only non-negative values (0,1,2,..)
nonPositiveInteger	An integer containing only non-positive values (..,-2,-1,0)
positiveInteger	An integer containing only positive values (1,2,..)
short	A signed 16-bit integer
unsignedLong	An unsigned 64-bit integer
unsignedInt	An unsigned 32-bit integer
unsignedShort	An unsigned 16-bit integer
unsignedByte	An unsigned 8-bit integer

Restrictions on Numeric Data Types
Restrictions that can be used with Numeric data types:

- enumeration
- fractionDigits
- maxExclusive
- maxInclusive
- minExclusive
- minInclusive
- pattern
- totalDigits
- whiteSpace

XSD Miscellaneous Data Types

Other miscellaneous data types are boolean, base64Binary, hexBinary, float, double, anyURI, QName, and NOTATION.

Boolean Data Type
The boolean data type is used to specify a true or false value.
The following is an example of a boolean declaration in a schema:

```
<xs:attribute name="disabled" type="xs:boolean"/>
```

An element in your document might look like this:

```
<prize disabled="true">999</prize>
```

Note: Legal values for boolean are true, false, 1 (which indicates true), and 0 (which indicates false).

Binary Data Types
Binary data types are used to express binary-formatted data.
We have two binary data types:

- base64Binary (Base64-encoded binary data)
- hexBinary (hexadecimal-encoded binary data)

The following is an example of a hexBinary declaration in a schema:

```
<xs:element name="blobsrc" type="xs:hexBinary"/>
```

AnyURI Data Type
The anyURI data type is used to specify a URI.
The following is an example of an anyURI declaration in a schema:

```
<xs:attribute name="src" type="xs:anyURI"/>
```

An element in your document might look like this:

```
<pic src="http://www.w3schools.com/images/smiley.gif" />
```

Note: If a URI has spaces, replace them with %20.

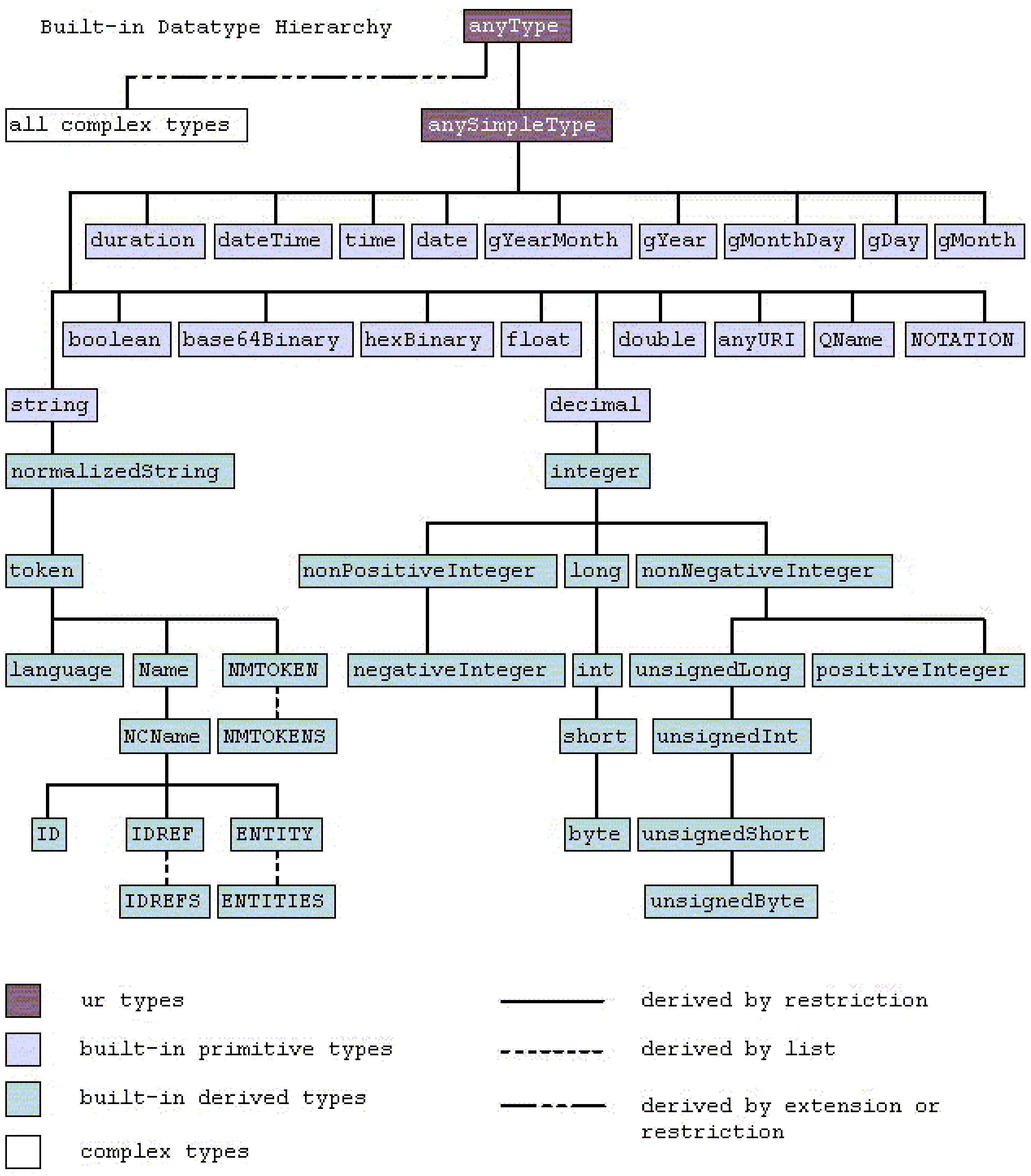
Miscellaneous Data Types

Name	Description
anyURI	
base64Binary	
boolean	
double	
float	
hexBinary	
NOTATION	
QName	

Restrictions on Miscellaneous Data Types
Restrictions that can be used with the other data types:

- enumeration (a Boolean data type cannot use this constraint)
- length (a Boolean data type cannot use this constraint)
- maxLength (a Boolean data type cannot use this constraint)
- minLength (a Boolean data type cannot use this constraint)
- pattern
- whiteSpace

Built-in XSD Data Types Hierarchy



XML Schema Reference

XSD Elements

Element	Explanation
all	Specifies that the child elements can appear in any order. Each child element can occur 0 or 1 time
annotation	Specifies the top-level element for schema comments
any	Enables the author to extend the XML document with elements not specified by the schema
anyAttribute	Enables the author to extend the XML document with attributes not specified by the schema
appInfo	Specifies information to be used by the application (must go inside annotation)
attribute	Defines an attribute
attributeGroup	Defines an attribute group to be used in complex type definitions
choice	Allows only one of the elements contained in the <choice> declaration to be present within the containing element
complexContent	Defines extensions or restrictions on a complex type that contains mixed content or elements only
complexType	Defines a complex type element
documentation	Defines text comments in a schema (must go inside annotation)
element	Defines an element
extension	Extends an existing simpleType or complexType element
field	Specifies an XPath expression that specifies the value used to define an identity constraint
group	Defines a group of elements to be used in complex type definitions
import	Adds multiple schemas with different target namespace to a document
include	Adds multiple schemas with the same target namespace to a document
key	Specifies an attribute or element value as a key (unique, non-nullable, and always present) within the containing element in an instance document
keyref	Specifies that an attribute or element value correspond to those of the specified key or unique element
list	Defines a simple type element as a list of values
notation	Describes the format of non-XML data within an XML document
redefine	Redefines simple and complex types, groups, and attribute groups from an external schema
restriction	Defines restrictions on a simpleType, simpleContent, or a complexContent
schema	Defines the root element of a schema
selector	Specifies an XPath expression that selects a set of elements for an identity constraint
sequence	Specifies that the child elements must appear in a sequence. Each child element can occur from 0 to any number of times
simpleContent	Contains extensions or restrictions on a text-only complex type or on a simple type as content and contains no elements
simpleType	Defines a simple type and specifies the constraints and information about the values of attributes or text-only elements
union	Defines a simple type as a collection (union) of values from specified simple data types
unique	Defines that an element or an attribute value must be unique within the

	scope

XSD Restrictions/Facets for Datatypes

Constraint	**Description**
enumeration	Defines a list of acceptable values
fractionDigits	Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero
length	Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero
maxExclusive	Specifies the upper bounds for numeric values (the value must be less than this value)
maxInclusive	Specifies the upper bounds for numeric values (the value must be less than or equal to this value)
maxLength	Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero
minExclusive	Specifies the lower bounds for numeric values (the value must be greater than this value)
minInclusive	Specifies the lower bounds for numeric values (the value must be greater than or equal to this value)
minLength	Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero
pattern	Defines the exact sequence of characters that are acceptable
totalDigits	Specifies the exact number of digits allowed. Must be greater than zero
whiteSpace	Specifies how white space (line feeds, tabs, spaces, and carriage returns) is handled