Schema documentation for book-schema-draft.xsd

november 15, 2014

Table of Contents

Namespace: "http://labs.rivendellweb.net/ns/books" 1
- Schema(s) 1
  - Main schema `book-schema-draft.xsd` 1
- Element(s) 2
  - Element `link` 2
  - Element `image` 2
  - Element `para` 4
  - Element `metadata` 5
  - Element `section` 6
  - Element `book` 7
- Simple Type(s) 8
  - Simple Type `string255` 8
  - Simple Type `isbn` 8
  - Simple Type `align` 8
- Attribute Group(s) 9
  - Attribute Group `genericPropertiesGroup` 9

Namespace: "" 9
- Element(s) 9
  - Element `para / strong` 9
  - Element `para / emphasis` 10
  - Element `para / underline` 10
  - Element `para / strike` 10
  - Element `para / span` 10
  - Element `metadata / isbn` 11
  - Element `metadata / edition` 11
  - Element `metadata / title` 11
  - Element `metadata / author` 11
  - Element `metadata / author / first-name` 12
  - Element `metadata / author / surname` 12
  - Element `metadata / editor` 12
  - Element `metadata / editor / first-name` 13
  - Element `metadata / editor / surname` 13
  - Element `metadata / editor / typeOfEditor` 14
  - Element `metadata / otherRole` 14
  - Element `metadata / otherRole / first-name` 14
  - Element `metadata / otherRole / last-name` 15
- Attribute(s) 15
  - Attribute `genericPropertiesGroup / @id` 15
  - Attribute `genericPropertiesGroup / @class` 15
  - Attribute `link / @href` 15
  - Attribute `link / @label` 16
  - Attribute `image / @src` 16
  - Attribute `image / @height` 16
  - Attribute `image / @width` 16
  - Attribute `image / @alt` 16
  - Attribute `image / @align` 17
  - Attribute `metadata / author / @id` 17
  - Attribute `metadata / editor / @id` 17
  - Attribute `metadata / otherRole / @id` 17
  - Attribute `section / @type` 17

Namespace: "http://labs.rivendellweb.net/ns/books"

Schema(s)

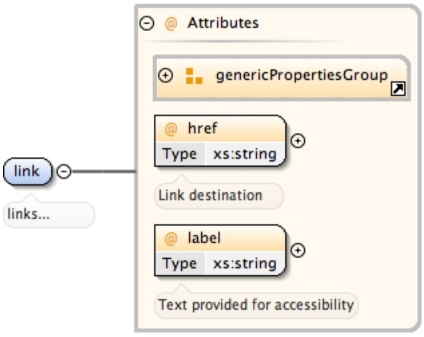
Main schema `book-schema-draft.xsd`

Namespace	http://labs.rivendellweb.net/ns/books

<table>
<tr><td>Properties</td><td><table><tr><td>attribute form default:</td><td>unqualified</td></tr><tr><td>element form default:</td><td>unqualified</td></tr></table></td></tr>
</table>

Element(s)

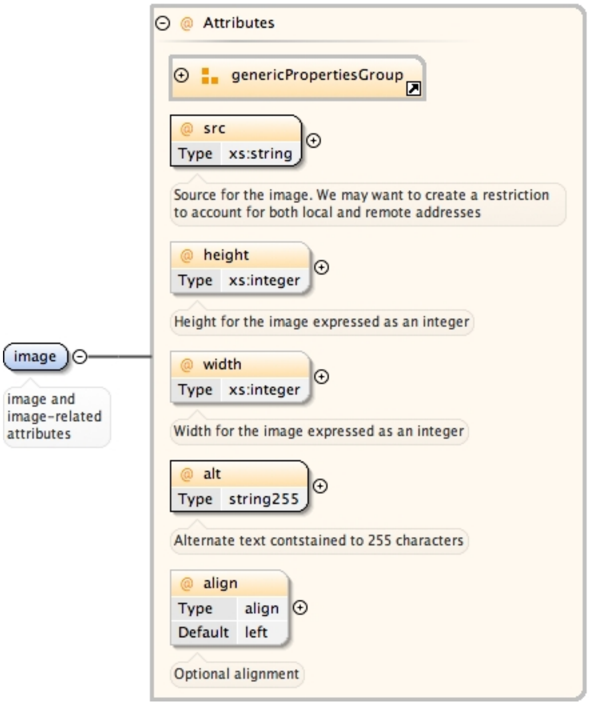
Element link

<table>
<tr><td>Namespace</td><td>http://labs.rivendellweb.net/ns/books</td></tr>
<tr><td>Annotations</td><td>links...</td></tr>
<tr><td>Diagram</td><td>
</td></tr>
<tr><td>Properties</td><td><table><tr><td>content:</td><td>complex</td></tr></table></td></tr>
<tr><td>Used by</td><td><table><tr><td>Element</td><td>para</td></tr></table></td></tr>
<tr><td>Attributes</td><td><table>
<tr><th>QName</th><th>Type</th><th>Use</th><th></th></tr>
<tr><td><b>class</b></td><td>xs:string</td><td>optional</td><td></td></tr>
<tr><td></td><td colspan="3">Class for the paragraph if any</td></tr>
<tr><td><b>href</b></td><td>xs:string</td><td>required</td><td></td></tr>
<tr><td></td><td colspan="3">Link destination</td></tr>
<tr><td><b>id</b></td><td>xs:ID</td><td>optional</td><td></td></tr>
<tr><td></td><td colspan="3">ID for the paragraph if any</td></tr>
<tr><td><b>label</b></td><td>xs:string</td><td>required</td><td></td></tr>
<tr><td></td><td colspan="3">Text provided for accessibility</td></tr>
</table></td></tr>
<tr><td>Source</td><td><pre>
<xs:element name="link">
  <xs:annotation>
    <xs:documentation>links...</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attributeGroup ref="genericPropertiesGroup"/>
    <xs:attribute name="href" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation>Link destination</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="label" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation>Text provided for accessibility</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
</pre></td></tr>
</table>

Element image

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	image and image-related attributes

Diagram


Attributes
genericPropertiesGroup
src
Type xs:string
Source for the image. We may want to create a restriction to account for both local and remote addresses
height
Type xs:integer
Height for the image expressed as an integer
image
image and image-related attributes
width
Type xs:integer
Width for the image expressed as an integer
alt
Type string255
Alternate text contstained to 255 characters
align
Type align
Default left
Optional alignment

Properties

content: `complex`

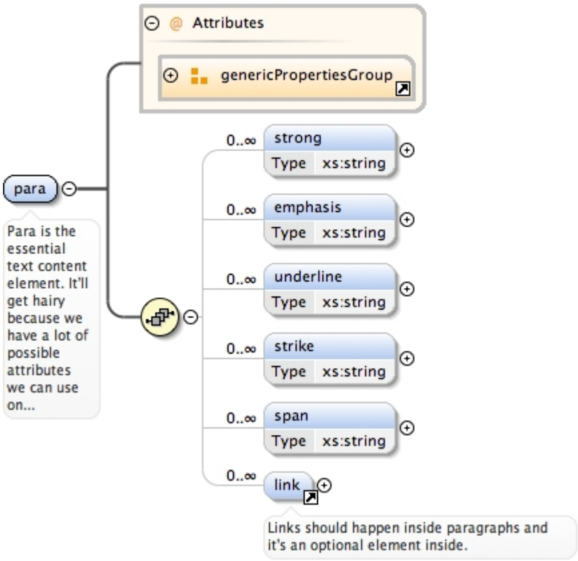
Attributes

QName	Type	Default	Use	
align	align	left	optional	
	Optional alignment			
alt	string255		required	
	Alternate text contstained to 255 characters			
class	xs:string		optional	
	Class for the paragraph if any			
height	xs:integer		optional	
	Height for the image expressed as an integer			
id	xs:ID		optional	
	ID for the paragraph if any			
src	xs:string		required	
	Source for the image. We may want to create a restriction to account for both local and remote addresses			
width	xs:integer		optional	
	Width for the image expressed as an integer			

Source

```
<xs:element name="image">
  <xs:annotation>
    <xs:documentation>image and image-related attributes</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:attributeGroup ref="genericPropertiesGroup"/>
    <xs:attribute name="src" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation>Source for the image. We may want to create a restriction to account for
 both local and remote addresses</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="height" type="xs:integer" use="optional">
      <xs:annotation>
        <xs:documentation>Height for the image expressed as an integer</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="width" type="xs:integer" use="optional">
      <xs:annotation>
        <xs:documentation>Width for the image expressed as an integer</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="alt" type="string255" use="required">
      <xs:annotation>
        <xs:documentation>Alternate text contstained to 255 characters</xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="align" type="align" use="optional" default="left">
      <xs:annotation>
        <xs:documentation>Optional alignment</xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element para

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	Para is the essential text content element. It'll get hairy because we have a lot of possible attributes we can use on it
Diagram	 Attributes genericPropertiesGroup para Para is the essential text content element. It'll get hairy because we have a lot of possible attributes we can use on... 0..∞ strong Type xs:string 0..∞ emphasis Type xs:string 0..∞ underline Type xs:string 0..∞ strike Type xs:string 0..∞ span Type xs:string 0..∞ link Links should happen inside paragraphs and it's an optional element inside.
Properties	content: `complex` mixed: `true`
Used by	Element section
Model	strong* , emphasis* , underline* , strike* , span* , link*
Children	emphasis, link, span, strike, strong, underline

Instance

```
<para class="" id="" xmlns="http://labs.rivendellweb.net/ns/books">
  <strong>{0,unbounded}</strong>
  <emphasis>{0,unbounded}</emphasis>
  <underline>{0,unbounded}</underline>
  <strike>{0,unbounded}</strike>
  <span>{0,unbounded}</span>
  <link class="" href="" id="" label="">{0,unbounded}</link>
</para>
```

Attributes

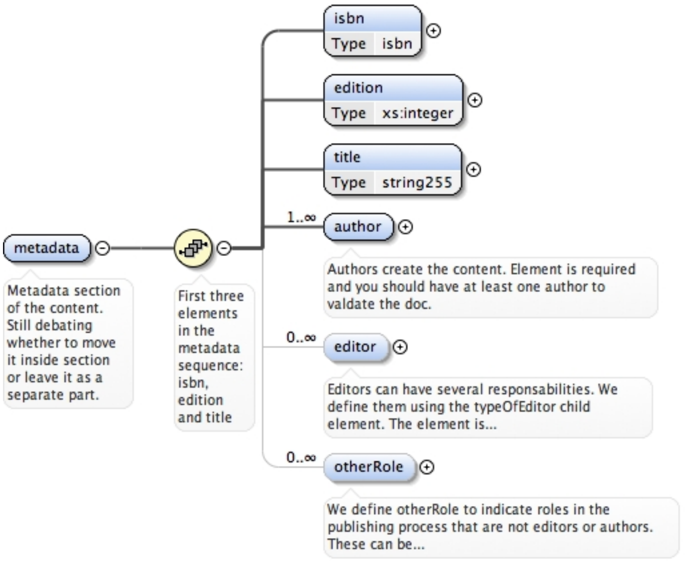
QName	Type	Use	
class	xs:string	optional	
	Class for the paragraph if any		
id	xs:ID	optional	
	ID for the paragraph if any		

Source

```
<xs:element name="para">
  <xs:annotation>
    <xs:documentation>Para is the essential text content element. It'll get hairy because we have a
 lot of possible attributes we can use on it</xs:documentation>
  </xs:annotation>
  <xs:complexType mixed="true">
    <xs:sequence>
      <!--
                    Style Elements.
```

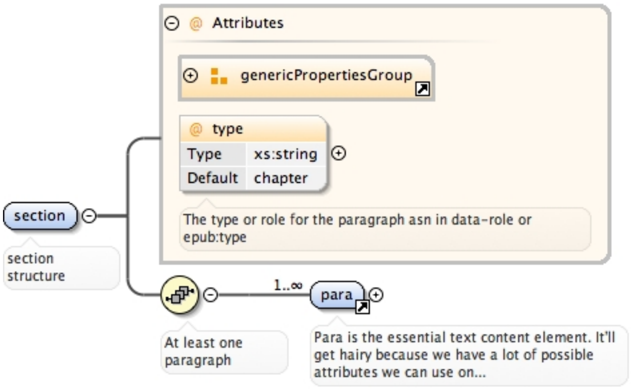
	We use strong and emphasis rather than bold and italics to try and stay in synch with HTML and HTML5. We may add additional tags later in the process. --> <xs:element name="strong" type="xs:string" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="emphasis" type="xs:string" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="underline" type="xs:string" minOccurs="0" maxOccurs="unbounded"/> <xs:element name="strike" type="xs:string" minOccurs="0" maxOccurs="unbounded"/> <!-- Organization Elements --> <xs:element name="span" type="xs:string" minOccurs="0" maxOccurs="unbounded"/> <xs:element ref="link" minOccurs="0" maxOccurs="unbounded"> <xs:annotation> <xs:documentation>Links should happen inside paragraphs and it's an optional element inside.</xs:documentation> </xs:annotation> </xs:element> </xs:sequence> <xs:attributeGroup ref="genericPropertiesGroup"/> </xs:complexType> </xs:element>

Element metadata

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	Metadata section of the content. Still debating whether to move it inside section or leave it as a separate part.
Diagram	 isbn Type isbn edition Type xs:integer title Type string255 1..∞ author 0..∞ editor 0..∞ otherRole metadata Metadata section of the content. Still debating whether to move it inside section or leave it as a separate part. First three elements in the metadata sequence: isbn, edition and title Authors create the content. Element is required and you should have at least one author to valdate the doc. Editors can have several responsabilities. We define them using the typeOfEditor child element. The element is... We define otherRole to indicate roles in the publishing process that are not editors or authors. These can be...
Properties	content: complex
Used by	Element book
Model	isbn , edition , title , author+ , editor* , otherRole*
Children	author, edition, editor, isbn, otherRole, title
Instance	<metadata xmlns="http://labs.rivendellweb.net/ns/books"> <isbn>{1,1}</isbn> <edition>{1,1}</edition> <title>{1,1}</title> <author id="">{1,unbounded}</author> <editor id="">{0,unbounded}</editor> <otherRole id="">{0,unbounded}</otherRole> </metadata>
Source	<xs:element name="metadata"> <xs:annotation> <xs:documentation>Metadata section of the content. Still debating whether to move it inside section or leave it as a separate part.</xs:documentation> </xs:annotation> <xs:complexType> <xs:sequence> <xs:annotation> <xs:documentation>First three elements in the metadata sequence: isbn, edition and title</xs:documentation> </xs:annotation>

```
      <xs:element name="isbn" type="isbn"/>
      <xs:element name="edition" type="xs:integer"/>
      <xs:element name="title" type="string255"/>
      <xs:element name="author" minOccurs="1" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>Authors create the content. Element is required and you should have at
 least one author to valdate the doc.</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="first-name" type="xs:string"/>
            <xs:element name="surname" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="id" type="xs:ID"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="editor" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>Editors can have several responsabilities. We define them using the
 typeOfEditor child element. The element is optional. If it's used we can have as many editors as we
 need</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="first-name" type="xs:string"/>
            <xs:element name="surname" type="xs:string"/>
            <xs:element name="typeOfEditor" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="id" type="xs:ID"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="otherRole" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation>We define otherRole to indicate roles in the publishing process that are
 not editors or authors. These can be reviewers, illustrators</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="first-name" type="xs:string"/>
            <xs:element name="last-name" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="id" type="xs:ID"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Element section

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	`section structure`
Diagram	 Attributes genericPropertiesGroup type Type xs:string Default chapter The type or role for the paragraph asn in data-role or epub:type section section structure 1..∞ para At least one paragraph Para is the essential text content element. It'll get hairy because we have a lot of possible attributes we can use on...
Properties	content: `complex` mixed: `true`
Used by	Element book
Model	para+
Children	para
Instance	`<section class="" id="" type="chapter" xmlns="http://labs.rivendellweb.net/ns/books">`

```
  <para class="" id="">{1,unbounded}</para>
</section>
```

Attributes

QName	Type	Default	Use	
class	xs:string		optional	
	Class for the paragraph if any			
id	xs:ID		optional	
	ID for the paragraph if any			
type	xs:string	chapter	optional	
	The type or role for the paragraph asn in data-role or epub:type			

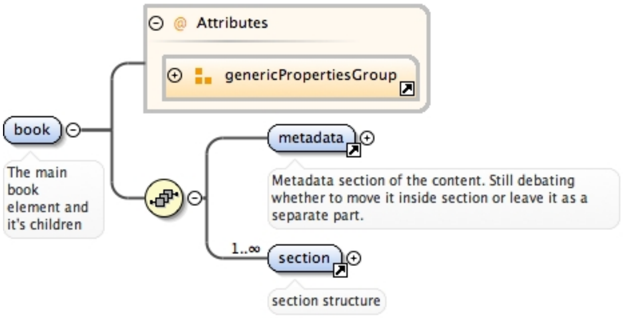
Source

```
<xs:element name="section">
  <xs:annotation>
    <xs:documentation>section structure</xs:documentation>
  </xs:annotation>
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:annotation>
        <xs:documentation>At least one paragraph</xs:documentation>
      </xs:annotation>
      <xs:element ref="para" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="genericPropertiesGroup"/>
    <xs:attribute name="type" type="xs:string" use="optional" default="chapter">
      <xs:annotation>
        <xs:documentation>The type or role for the paragraph asn in data-role or epub:type</
xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
</xs:element>
```

Element book

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	The main book element and it's children

Diagram


Attributes
genericPropertiesGroup
book
The main book element and it's children
metadata
Metadata section of the content. Still debating whether to move it inside section or leave it as a separate part.
1..∞
section
section structure

Properties

content:	complex
mixed:	true

Model	metadata , section+
Children	metadata, section

Instance

```
<book class="" id="" xmlns="http://labs.rivendellweb.net/ns/books">
  <metadata>{1,1}</metadata>
  <section class="" id="" type="chapter">{1,unbounded}</section>
</book>
```

Attributes

QName	Type	Use	
class	xs:string	optional	
	Class for the paragraph if any		
id	xs:ID	optional	
	ID for the paragraph if any		

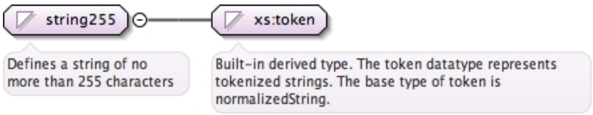
Source

```
<xs:element name="book">
  <xs:annotation>
    <xs:documentation>The main book element and it's children</xs:documentation>
  </xs:annotation>
  <xs:complexType mixed="true">
    <xs:annotation>
```

```
      <xs:documentation>A sequence of one metadat section followed by 1 or more sections</
xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element ref="metadata" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="section" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="genericPropertiesGroup"/>
  </xs:complexType>
</xs:element>
```

Simple Type(s)

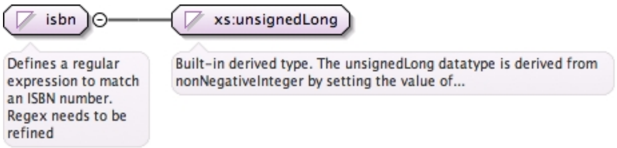
Simple Type string255

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	Defines a string of no more than 255 characters
Diagram	 Defines a string of no more than 255 characters Built-in derived type. The token datatype represents tokenized strings. The base type of token is normalizedString.
Type	restriction of xs:token
Facets	maxLength 255
Used by	Attribute image/@alt Element metadata/title

Source

```
<xs:simpleType name="string255">
  <xs:annotation>
    <xs:documentation>Defines a string of no more than 255 characters</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:token">
    <xs:maxLength value="255"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type isbn

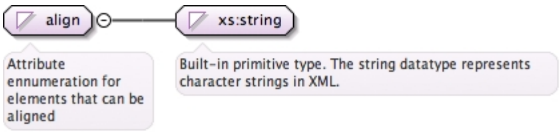
Namespace	http://labs.rivendellweb.net/ns/books
Annotations	Defines a regular expression to match an ISBN number. Regex needs to be refined
Diagram	 Defines a regular expression to match an ISBN number. Regex needs to be refined Built-in derived type. The unsignedLong datatype is derived from nonNegativeInteger by setting the value of...
Type	restriction of xs:unsignedLong
Facets	totalDigits 10 pattern ([\-+]?[0-9]+) & (\d{10})
Used by	Element metadata/isbn

Source

```
<xs:simpleType name="isbn">
  <xs:annotation>
    <xs:documentation>Defines a regular expression to match an ISBN number. Regex needs to be
 refined</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:unsignedLong">
    <xs:totalDigits value="10"/>
    <xs:pattern value="\d{10}"/>
  </xs:restriction>
</xs:simpleType>
```

Simple Type align

Namespace	http://labs.rivendellweb.net/ns/books
Annotations	Attribute ennumeration for elements that can be aligned

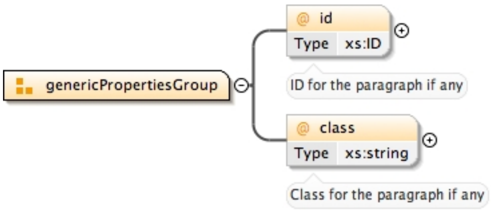
Diagram	 align xs:string Attribute ennumeration for elements that can be aligned Built-in primitive type. The string datatype represents character strings in XML.	
Type	restriction of xs:string	
Facets	enumeration	`left`
	enumeration	`center`
	enumeration	`right`
	enumeration	`justify`
Used by	Attribute	image/@align

Source

```
<xs:simpleType name="align">
  <xs:annotation>
    <xs:documentation>Attribute ennumeration for elements that can be aligned</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="left"/>
    <xs:enumeration value="center"/>
    <xs:enumeration value="right"/>
    <xs:enumeration value="justify"/>
  </xs:restriction>
</xs:simpleType>
```

Attribute Group(s)

Attribute Group `genericPropertiesGroup`

Namespace	http://labs.rivendellweb.net/ns/books	
Diagram	 genericPropertiesGroup @ id Type xs:ID ID for the paragraph if any @ class Type xs:string Class for the paragraph if any	
Used by	Elements	book, image, link, para, section

Attributes

QName	Type	Use	
class	xs:string	optional	
	`Class for the paragraph if any`		
id	xs:ID	optional	
	`ID for the paragraph if any`		

Source

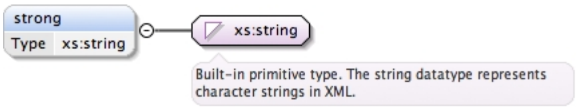
```
<xs:attributeGroup name="genericPropertiesGroup">
  <xs:attribute name="id" type="xs:ID" use="optional">
    <xs:annotation>
      <xs:documentation>ID for the paragraph if any</xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="class" type="xs:string" use="optional">
    <xs:annotation>
      <xs:documentation>Class for the paragraph if any</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:attributeGroup>
```

Namespace: ""

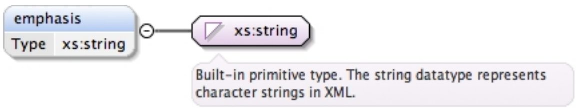
Element(s)

Element `para / strong`

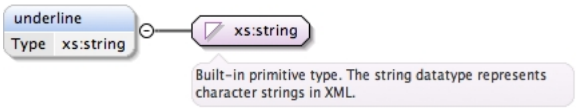
Namespace	No namespace

Diagram	 strong Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple minOccurs: 0 maxOccurs: unbounded
Source	`<xs:element name="strong" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>`

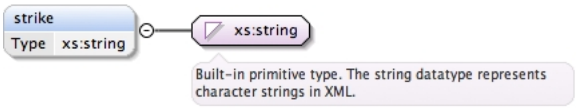
Element para / emphasis

Namespace	No namespace
Diagram	 emphasis Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple minOccurs: 0 maxOccurs: unbounded
Source	`<xs:element name="emphasis" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>`

Element para / underline

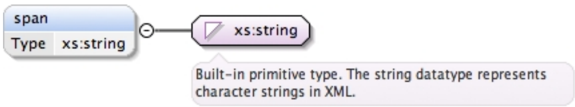
Namespace	No namespace
Diagram	 underline Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple minOccurs: 0 maxOccurs: unbounded
Source	`<xs:element name="underline" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>`

Element para / strike

Namespace	No namespace
Diagram	 strike Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple minOccurs: 0 maxOccurs: unbounded
Source	`<xs:element name="strike" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>`

Element para / span

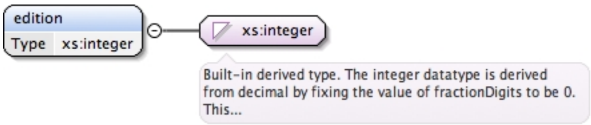
Namespace	No namespace

Diagram	 span Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: `simple` minOccurs: `0` maxOccurs: `unbounded`
Source	`<xs:element name="span" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>`

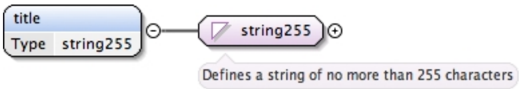
Element metadata / isbn

Namespace	No namespace
Diagram	isbn Type isbn isbn Defines a regular expression to match an ISBN number. Regex needs to be refined
Type	isbn
Properties	content: `simple`
Facets	totalDigits `10` pattern `([\-+]?[0-9]+) & (\d{10})`
Source	`<xs:element name="isbn" type="isbn"/>`

Element metadata / edition

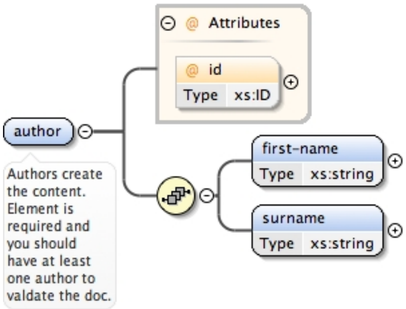
Namespace	No namespace
Diagram	 edition Type xs:integer xs:integer Built-in derived type. The integer datatype is derived from decimal by fixing the value of fractionDigits to be 0. This...
Type	xs:integer
Properties	content: `simple`
Source	`<xs:element name="edition" type="xs:integer"/>`

Element metadata / title

Namespace	No namespace
Diagram	 title Type string255 string255 Defines a string of no more than 255 characters
Type	string255
Properties	content: `simple`
Facets	maxLength `255`
Source	`<xs:element name="title" type="string255"/>`

Element metadata / author

Namespace	No namespace
Annotations	`Authors create the content. Element is required and you should have at least one author to valdate the doc.`

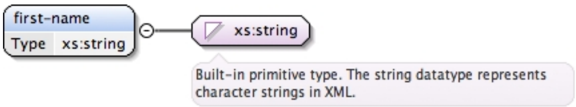
Diagram	 Attributes id Type xs:ID author Authors create the content. Element is required and you should have at least one author to valdate the doc. first-name Type xs:string surname Type xs:string
Properties	content: complex minOccurs: 1 maxOccurs: unbounded
Model	first-name , surname
Children	first-name, surname

Instance	<author id="">
	  <first-name>{1,1}</first-name>
	  <surname>{1,1}</surname>
	</author>

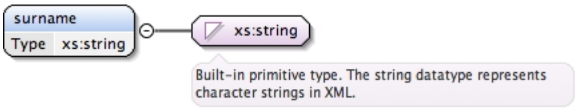
Attributes	**QName** | **Type** | **Use** **id** | xs:ID | optional

Source	<xs:element name="author" minOccurs="1" maxOccurs="unbounded">
	  <xs:annotation>
	    <xs:documentation>Authors create the content. Element is required and you should have at least one author to valdate the doc.</xs:documentation>
	  </xs:annotation>
	  <xs:complexType>
	    <xs:sequence>
	      <xs:element name="first-name" type="xs:string"/>
	      <xs:element name="surname" type="xs:string"/>
	    </xs:sequence>
	    <xs:attribute name="id" type="xs:ID"/>
	  </xs:complexType>
	</xs:element>

Element metadata / author / first-name

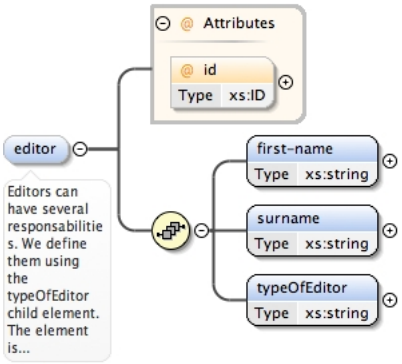
Namespace	No namespace
Diagram	 first-name Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	<xs:element name="first-name" type="xs:string"/>

Element metadata / author / surname

Namespace	No namespace
Diagram	 surname Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	<xs:element name="surname" type="xs:string"/>

Element metadata / editor

Namespace	No namespace

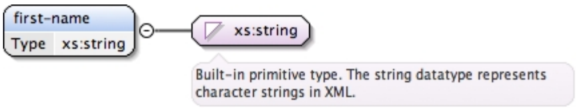
Annotations	Editors can have several responsabilities. We define them using the typeOfEditor child element. The element is optional. If it's used we can have as many editors as we need
Diagram	 Attributes id Type xs:ID editor Editors can have several responsabilitie s. We define them using the typeOfEditor child element. The element is... first-name Type xs:string surname Type xs:string typeOfEditor Type xs:string
Properties	content: complex minOccurs: 0 maxOccurs: unbounded
Model	first-name , surname , typeOfEditor
Children	first-name, surname, typeOfEditor

Instance	`<editor id="">`
	`  <first-name>{1,1}</first-name>`
	`  <surname>{1,1}</surname>`
	`  <typeOfEditor>{1,1}</typeOfEditor>`
	`</editor>`

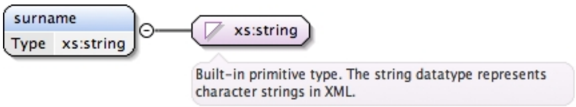
Attributes	**QName** | **Type** | **Use** **id** | xs:ID | optional

Source	`<xs:element name="editor" minOccurs="0" maxOccurs="unbounded">`
	`  <xs:annotation>`
	`    <xs:documentation>Editors can have several responsabilities. We define them using the typeOfEditor child element. The element is optional. If it's used we can have as many editors as we need</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:complexType>`
	`    <xs:sequence>`
	`      <xs:element name="first-name" type="xs:string"/>`
	`      <xs:element name="surname" type="xs:string"/>`
	`      <xs:element name="typeOfEditor" type="xs:string"/>`
	`    </xs:sequence>`
	`    <xs:attribute name="id" type="xs:ID"/>`
	`  </xs:complexType>`
	`</xs:element>`

Element metadata / editor / first-name

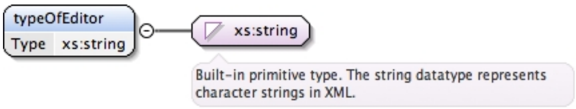
Namespace	No namespace
Diagram	 first-name Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	`<xs:element name="first-name" type="xs:string"/>`

Element metadata / editor / surname

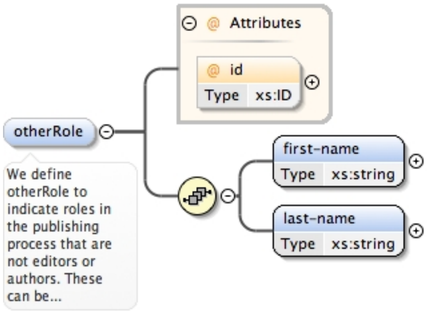
Namespace	No namespace
Diagram	 surname Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.

Type	xs:string
Properties	content: simple
Source	`<xs:element name="surname" type="xs:string"/>`

Element metadata / editor / typeOfEditor

Namespace	No namespace
Diagram	 typeOfEditor Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	`<xs:element name="typeOfEditor" type="xs:string"/>`

Element metadata / otherRole

Namespace	No namespace
Annotations	We define otherRole to indicate roles in the publishing process that are not editors or authors. These can be reviewers, illustrators
Diagram	 Attributes id Type xs:ID otherRole We define otherRole to indicate roles in the publishing process that are not editors or authors. These can be... first-name Type xs:string last-name Type xs:string
Properties	content: complex minOccurs: 0 maxOccurs: unbounded
Model	first-name , last-name
Children	first-name, last-name

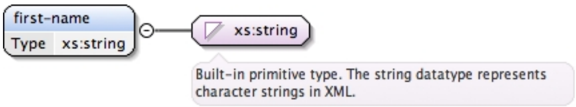
Instance	`<otherRole id="">`
	`  <first-name>{1,1}</first-name>`
	`  <last-name>{1,1}</last-name>`
	`</otherRole>`

Attributes	**QName** | **Type** | **Use** **id** | xs:ID | optional

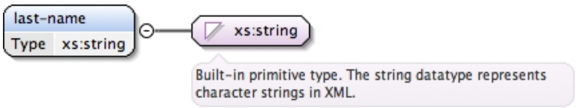
Source	`<xs:element name="otherRole" minOccurs="0" maxOccurs="unbounded">`
	`  <xs:annotation>`
	`    <xs:documentation>We define otherRole to indicate roles in the publishing process that are not editors or authors. These can be reviewers, illustrators</xs:documentation>`
	`  </xs:annotation>`
	`  <xs:complexType>`
	`    <xs:sequence>`
	`      <xs:element name="first-name" type="xs:string"/>`
	`      <xs:element name="last-name" type="xs:string"/>`
	`    </xs:sequence>`
	`    <xs:attribute name="id" type="xs:ID"/>`
	`  </xs:complexType>`
	`</xs:element>`

Element metadata / otherRole / first-name

Namespace	No namespace

Diagram	 first-name Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	`<xs:element name="first-name" type="xs:string"/>`

Element metadata / otherRole / last-name

Namespace	No namespace
Diagram	 last-name Type xs:string xs:string Built-in primitive type. The string datatype represents character strings in XML.
Type	xs:string
Properties	content: simple
Source	`<xs:element name="last-name" type="xs:string"/>`

Attribute(s)

Attribute genericPropertiesGroup / @id

Namespace	No namespace
Annotations	ID for the paragraph if any
Type	xs:ID
Properties	use: optional
Used by	Attribute Group genericPropertiesGroup

Source	`<xs:attribute name="id" type="xs:ID" use="optional">`
	`  <xs:annotation>`
	`    <xs:documentation>ID for the paragraph if any</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute genericPropertiesGroup / @class

Namespace	No namespace
Annotations	Class for the paragraph if any
Type	xs:string
Properties	use: optional
Used by	Attribute Group genericPropertiesGroup

Source	`<xs:attribute name="class" type="xs:string" use="optional">`
	`  <xs:annotation>`
	`    <xs:documentation>Class for the paragraph if any</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute link / @href

Namespace	No namespace
Annotations	Link destination
Type	xs:string
Properties	use: required
Used by	Element link

Source	`<xs:attribute name="href" type="xs:string" use="required">`
	`  <xs:annotation>`
	`    <xs:documentation>Link destination</xs:documentation>`
	`  </xs:annotation>`

	</xs:attribute>

Attribute link / @label

Namespace	No namespace
Annotations	Text provided for accessibility
Type	xs:string
Properties	use: required
Used by	Element link
Source	<xs:attribute name="label" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Text provided for accessibility</xs:documentation> </xs:annotation> </xs:attribute>

Attribute image / @src

Namespace	No namespace
Annotations	Source for the image. We may want to create a restriction to account for both local and remote addresses
Type	xs:string
Properties	use: required
Used by	Element image
Source	<xs:attribute name="src" type="xs:string" use="required"> <xs:annotation> <xs:documentation>Source for the image. We may want to create a restriction to account for both local and remote addresses</xs:documentation> </xs:annotation> </xs:attribute>

Attribute image / @height

Namespace	No namespace
Annotations	Height for the image expressed as an integer
Type	xs:integer
Properties	use: optional
Used by	Element image
Source	<xs:attribute name="height" type="xs:integer" use="optional"> <xs:annotation> <xs:documentation>Height for the image expressed as an integer</xs:documentation> </xs:annotation> </xs:attribute>

Attribute image / @width

Namespace	No namespace
Annotations	Width for the image expressed as an integer
Type	xs:integer
Properties	use: optional
Used by	Element image
Source	<xs:attribute name="width" type="xs:integer" use="optional"> <xs:annotation> <xs:documentation>Width for the image expressed as an integer</xs:documentation> </xs:annotation> </xs:attribute>

Attribute image / @alt

Namespace	No namespace
Annotations	Alternate text contstained to 255 characters

Type	string255
Properties	use: required
Facets	maxLength 255
Used by	Element image

Source	`<xs:attribute name="alt" type="string255" use="required">`
	`  <xs:annotation>`
	`    <xs:documentation>Alternate text contstained to 255 characters</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute image / @align

Namespace	No namespace
Annotations	Optional alignment
Type	align
Properties	use: optional default: left
Facets	enumeration left enumeration center enumeration right enumeration justify
Used by	Element image

Source	`<xs:attribute name="align" type="align" use="optional" default="left">`
	`  <xs:annotation>`
	`    <xs:documentation>Optional alignment</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`

Attribute metadata / author / @id

Namespace	No namespace
Type	xs:ID
Properties	content: simple
Used by	Element metadata/author
Source	`<xs:attribute name="id" type="xs:ID"/>`

Attribute metadata / editor / @id

Namespace	No namespace
Type	xs:ID
Properties	content: simple
Used by	Element metadata/editor
Source	`<xs:attribute name="id" type="xs:ID"/>`

Attribute metadata / otherRole / @id

Namespace	No namespace
Type	xs:ID
Properties	content: simple
Used by	Element metadata/otherRole
Source	`<xs:attribute name="id" type="xs:ID"/>`

Attribute section / @type

Namespace	No namespace

Annotations	The type or role for the paragraph asn in data-role or epub:type
Type	xs:string
Properties	use: optional default: chapter
Used by	Element section

Source	`<xs:attribute name="type" type="xs:string" use="optional" default="chapter">`
	`  <xs:annotation>`
	`    <xs:documentation>The type or role for the paragraph asn in data-role or epub:type</xs:documentation>`
	`  </xs:annotation>`
	`</xs:attribute>`