

XSL Library Template Reference

Copyright © 1999, 2000, 2001, 2002 Norman Walsh

COLLABORATORS			
	TITLE : XSL Library Template Reference		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		March 28, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1	General Library Templates	1
1.1	dot.count	1
1.2	copy-string	1
1.3	string.subst	2
1.4	xpointer.idref	2
1.5	length-magnitude	3
1.6	length-units	3
1.7	length-spec	4
1.8	length-in-points	5
1.9	pi-attribute	6
1.10	lookup.key	6
1.11	xpath.location	7
1.12	comment-escape-string	8
1.13	comment-escape-string.recursive	8
2	Relative URI Functions	9
2.1	count.uri.path.depth	9
2.2	trim.common.uri.paths	9
A	The Stylesheet	11

Introduction

This is technical reference documentation for the DocBook XSL Stylesheets; it documents (some of) the parameters, templates, and other elements of the stylesheets.

This is not intended to be “user” documentation. It is provided for developers writing customization layers for the stylesheets, and for anyone who’s interested in “how it works”.

Although I am trying to be thorough, this documentation is known to be incomplete. Don’t forget to read the source, too :-)

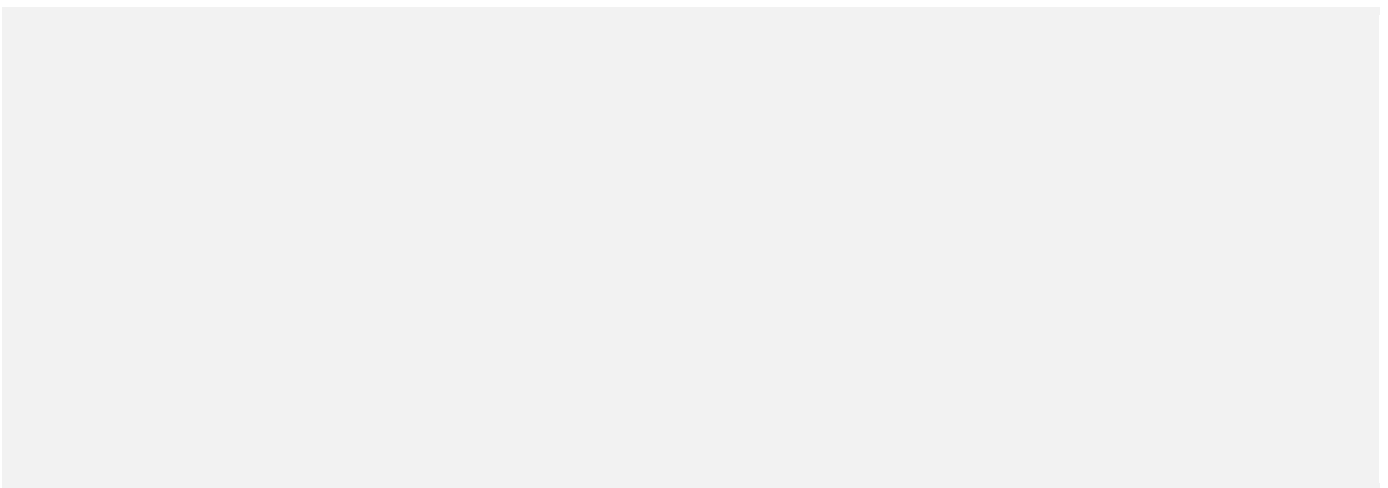
Chapter 1

General Library Templates

1.1 dot.count

dot.count — Returns the number of “.” characters in a string

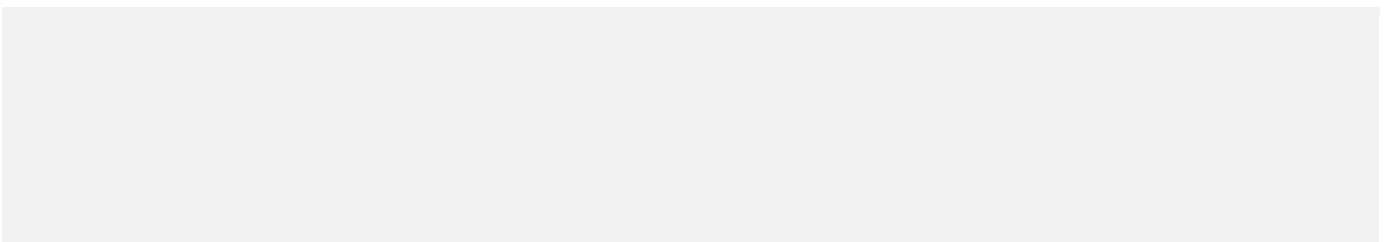
Description



1.2 copy-string

copy-string — Returns “count” copies of a string

Description



1.3 string.subst

string.subst — Substitute one text string for another in a string

Description

The `string.subst` template replaces all occurrences of *target* in *string* with *replacement* and returns the result.

1.4 xpointer.idref

xpointer.idref — Extract IDREF from an XPointer

Description

The `xpointer.idref` template returns the ID portion of an XPointer which is a pointer to an ID within the current document, or the empty string if it is not.

In other words, `xpointer.idref` returns “foo” when passed either `#foo` or `#xpointer(id('foo'))`, otherwise it returns the empty string.

```
http://...
```

1.5 length-magnitude

length-magnitude — Return the unqualified dimension from a length specification

Description

The `length-magnitude` template returns the unqualified length ("20" for "20pt") from a dimension.

1.6 length-units

length-units — Return the units from a length specification

Description

The `length-units` template returns the units ("pt" for "20pt") from a length. If no units are supplied on the length, the *default.units* are returned.

1.7 length-spec

length-spec — Return a fully qualified length specification

Description

The `length-spec` template returns the qualified length from a dimension. If an unqualified length is given, the *default.units* will be added to it.

Unrecognized unit of measure:

.

1.8 length-in-points

length-in-points — Returns the size, in points, of a specified length

Description

The `length-in-points` template converts a length specification to points and returns that value as an unqualified number.



Caution

There is no way for the template to infer the size of an `em`. It relies on the default `em.size` which is initially 10 (for 10pt).

Similarly, converting pixels to points relies on the `pixels.per.inch` parameter which is initially 90.

Unrecognized unit of measure:

.

1.9 pi-attribute

pi-attribute — Extract a pseudo-attribute from a PI

Description

The `pi-attribute` template extracts a pseudo-attribute from a processing instruction. For example, given the PI “`<?foo bar="1" baz='red' ?>`”,

```
<xsl:call-template name="pi-attribute">
  <xsl:with-param name="pis" select="processing-instruction('foo')"/>
  <xsl:with-param name="attribute" select="'baz'"/>
</xsl:call-template>
```

will return “red”. This template returns the first matching attribute that it finds. Presented with processing instructions that contain badly formed pseudo-attributes (missing or unbalanced quotes, for example), the template may silently return erroneous results.

```
filename
1
```

1.10 lookup.key

lookup.key — Retrieve the value associated with a particular key in a table

Description

Given a table of space-delimited key/value pairs, the `lookup.key` template extracts the value associated with a particular key.

1.11 `xpath.location`

`xpath.location` — Calculate the XPath child-sequence to the current node

Description

The `xpath.location` template calculates the absolute path from the root of the tree to the current element node.

/

/

1.12 `comment-escape-string`

`comment-escape-string` — Prepare a string for inclusion in an XML comment

Description

The `comment-escape-string` template returns a string that has been transformed so that it can safely be output as an XML comment. Internal occurrences of "--" will be replaced with "- -" and a leading and/or trailing space will be added to the string, if necessary.

1.13 `comment-escape-string.recursive`

`comment-escape-string.recursive` — Internal function used by `comment-escape-string`

Description

The `comment-escape-string.recursive` template is used by `comment-escape-string`.

Chapter 2

Relative URI Functions

These functions manipulate relative URI references.

The following assumptions must hold true:

1. All URIs are relative.
2. No URI contains the “. ./” sequence which would effectively move “up” the hierarchy.

If these assumptions do not hold, the results are unpredictable.

2.1 `count.uri.path.depth`

`count.uri.path.depth` — Count the number of path components in a relative URI

Description

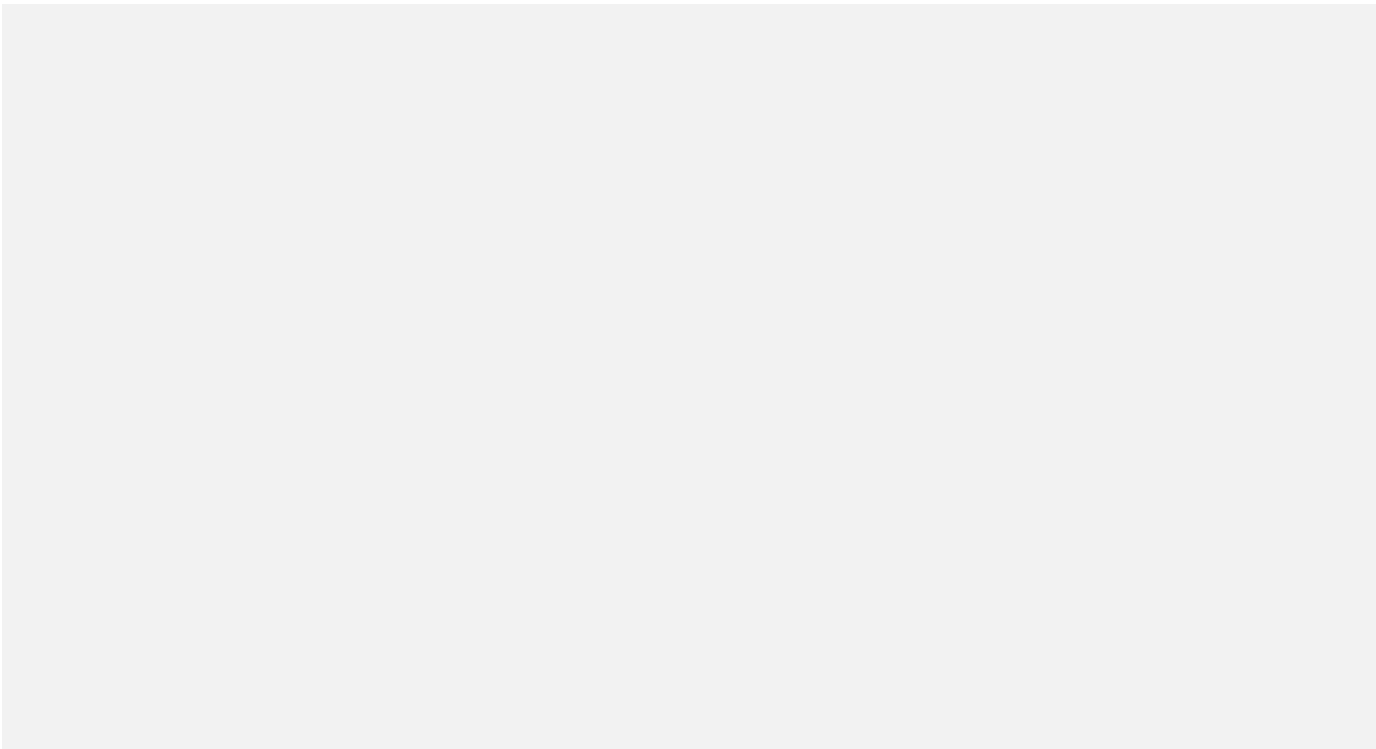
This function counts the number of path components in a relative URI.

2.2 `trim.common.uri.paths`

`trim.common.uri.paths` — Trim common leading path components from a relative URI

Description

This function trims common leading path components from a relative URI.



Appendix A

The Stylesheet

The `lib.xsl` stylesheet is just a wrapper around these functions.