xmllint XML check and query

CONCEPTS

With no flags xmllint parses the file and reprints it; add --noout when you only want the exit code.

Parse well-formed?
↓ --xinclude
Process XInclude, entities
↓ --valid / --schema
Validate DTD, XSD, RNG
↓ --xpath / --format
Output tree, nodes, c14n

Exit code tells the story: 0 ok, 3 invalid, 4 not well-formed, 10 XPath error, 11 XPath empty.

CHECK AND FORMAT

xmllint *<file>*
Parse and print the document

xmllint --noout *<file>*
Check well-formedness only

xmllint --format *<file>*
Reindent output

xmllint --format *<file>* **--output** *<out>*
Write result to a file

xmllint --format - < *<file>*
Read from stdin

curl -s *<url>* **| xmllint --format -**
Pretty-print a download

XMLLINT_INDENT=" " xmllint --format *<file>*
Choose the indent string

XPATH QUERIES

xmllint --xpath "//book" *<file>*
Select all matching nodes

xmllint --xpath "//book/title/text()" *<file>*
Get text content

xmllint --xpath "string(//book[1]/title)" *<file>*
Get one value as a string

xmllint --xpath "count(//book)" *<file>*
Count nodes

xmllint --xpath "//book/@id" *<file>*
Select attributes

XPATH QUERIES (cont.)

xmllint --xpath "//book[@id='b2']" *<file>*
Filter by attribute

xmllint --xpath "//book[price>10]/title" *<file>*
Filter by child value

xmllint --xpath "//*[local-name()='item']" *<file>*
Match ignoring namespaces

xmllint --xpath "boolean(//book)" *<file>*
Test existence

xmllint --xpath "//book[last()]" *<file>*
Pick the last node

xmllint --xpath "//book[position()⩽3]" *<file>*
Pick the first three

A node-set result prints as XML, a string or number as bare text. Wrap with string() for a clean value.

An empty result prints 'XPath set is empty' and exits 11, so scripts can test with if.

XPath quick reference

Syntax	Meaning
`/a/b`	child b of root a
`//b`	b anywhere
`.  ..`	this node, parent
`@id`	attribute id
`b[2]`	second b
`b[@id='x']`	b with id x
`text()`	text children
`*`	any element
`a/following-sibling::b`	later sibling b
`contains(@c,'x')`	substring test
`starts-with(n,'x')`	prefix test
`normalize-space(n)`	trim whitespace

VALIDATION

xmllint --noout --schema *<xsd> <file>*
Validate against an XSD

xmllint --noout --relaxng *<rng> <file>*
Validate against RelaxNG

VALIDATION (cont.)

xmllint --noout --dtdvalid *<dtd> <file>*
Validate against a DTD file

xmllint --noout --valid *<file>*
Validate with the document's own DOCTYPE

xmllint --noout --schematron *<sch> <file>*
Validate against Schematron

xmllint --noout --schema *<xsd> <file1> <file2>*
Check several files at once

xmllint --noout --stream --valid *<file>*
Validate big files with low memory

xmllint --noout --schema *<xsd> <file>* **&& echo ok**
Use the exit code in scripts

Validation success prints '<file> validates' to stderr; failures return exit 3 (or 5 if the schema itself is broken).

PARSING OPTIONS

xmllint --recover *<file>*
Salvage the parsable parts

xmllint --nonet *<file>*
Never fetch over the network

xmllint --xinclude *<file>*
Resolve XInclude elements

xmllint --noent *<file>*
Substitute entity references

xmllint --huge *<file>*
Lift built-in parser limits

HTML, CANONICAL, SHELL

xmllint --html *<file>*
Parse as HTML

xmllint --html --format *<file>*
Tidy up HTML

xmllint --html --xmlout *<file>*
Convert HTML to XHTML-style XML

xmllint --html --xpath "//a/@href" *<file>*
Scrape links from HTML