NAME

xo - emit formatted output based on format string and arguments

SYNOPSIS

xo [**-options**] [*argument...*]

DESCRIPTION

The **xo** utility allows command line access to the functionality of the **libxo** library. Using **xo**, shell scripts can emit *XML*, *JSON*, or *HTML* using the same commands that emit text output.

--close *path*
Close tags for the given path

-C | **--continuation**
Indicates this output is a continuation of the previous output data and should appear on the same line. This is allows HTML output to be constructed correctly.

--depth *num*
Set the depth for pretty printing

--help Display help text

-H | **--html**
Generate HTML output

-J | **--json**
Generate JSON output

--leading-xpath *path*
Add a prefix to generated XPaths (HTML)

--not-first
Indicate that this content is not the first in a series of sibling objects, which is vital information for "JSON" output, which requires a comma between such objects.

--open *path*
Open tags for the given path

-p | **--pretty**
Make ’pretty’ output (add indent, newlines)

--style *style*
Generate given style (xml, json, text, html)

-T | **--text**
Generate text output (the default style)

--top-warp
Indicates the entire object should be placed inside a top-level object wrapper, specifically when generating JSON output.

--version
Display version information

-W | **--warn**
Display warnings in text on stderr

--warn-xml
Display warnings in xml on stdout

--wrap *path*
Wrap output in a set of containers

-X | **--xml**
Generate XML output

--xpath
Add XPath data to HTML output

The **xo** utility accepts a format string suitable for xo_emit(3) and a set of zero or more arguments used to supply data for that string.

In addition, **xo** accepts any of the **libxo** options listed in xo_options(7).

EXAMPLES

In this example, **xo** is used to emit the same data encoded in text and then in XML by adding the "-p" (pretty) and "-X" (XML output) flags:

```
% xo 'The {:product} is {:status}\n' stereo "in route"
The stereo is in route
% xo -p -X 'The {:product} is {:status}\n' stereo "in route"
```

```
<product>stereo</product>
<status>in route</status>
```

In this example, the output from a **xo** command is shown in several styles:

```
xo "The {k:name} weighs {:weight/%d} pounds.\n" fish 6

TEXT:
  The fish weighs 6 pounds.
XML:
  <name>fish</name>
  <weight>6</weight>
JSON:
  "name": "fish",
  "weight": 6
HTML:
  <div class="line">
    <div class="text">The </div>
    <div class="data" data-tag="name">fish</div>
    <div class="text"> weighs </div>
    <div class="data" data-tag="weight">6</div>
    <div class="text"> pounds.</div>
  </div>
```

The **--wrap <path>** option can be used to wrap emitted content in a specific hierarchy. The path is a set of hierarchical names separated by the ’/’ character.

```
xo --wrap top/a/b/c ’{:tag}’ value

XML:
  <top>
    <a>
      <b>
        <c>
          <tag>value</tag>
        </c>
      </b>
    </a>
  </top>
JSON:
```

```
 "top": {
  "a": {
   "b": {
    "c": {
     "tag": "value"
    }
   }
  }
 }
```

The **--open <path>** and **--close <path>** can be used to emit hierarchical information without the matching close and open tag. This allows a shell script to emit open tags, data, and then close tags. The **--depth** option may be used to set the depth for indentation. The **--leading-xpath** may be used to prepend data to the XPath values used for HTML output style.

```
#!/bin/sh
xo --open top/data
xo --depth 2 '{:tag}' value
xo --close top/data

XML:
 <top>
  <data>
   <tag>value</tag>
  </data>
 </top>
JSON:
 "top": {
  "data": {
   "tag": "value"
  }
 }
```

SEE ALSO

libxo(3), xo_emit(3), xo_options(7)

HISTORY

The **libxo** library first appeared in FreeBSD 11.0.

AUTHORS

libxo was written by Phil Shafer <*phil@freebsd.org*>.