Home
Readme
Charter
Release Info
Installation
Download
Bug-Reporting
FAQs
Samples
API JavaDoc
Features
Properties
XNI Manual
XML Schema
SAX
DOM
Limitations
Source Repository
User Mail Archive
Dev Mail Archive

Overview

This page documents the various Document Object Model (DOM) samples included with Xerces. Besides being useful programs, they can be used as DOM programming examples to learn how to program using the DOM API.

Basic DOM samples:

- dom.Counter
- dom.GetElementsByTagName
- dom.Writer
- dom.ElementPrinter
- dom.DOM3
- dom.DOMAddLines

Most of the DOM parser samples have a command line option that allows the user to specify a different DOM parser to use. In order to supply another DOM parser besides the default Xerces `DOMParser`, a DOM parser wrapper class must be written. This class must implement the `dom.ParserWrapper` interface.

JAXP could be used instead of the special DOM parser wrapper class. However, that feature is not implemented at this time. Using JAXP would require the user to specify the `-Djavax.xml.parsers.DocumentBuilderFactory=...` *option to the virtual machine in order to use a different document builder factory.*

Sample dom.Counter

A sample DOM counter. This sample program illustrates how to traverse a DOM tree in order to get information about the document. The output of this program shows the time and count of elements, attributes, ignorable whitespaces, and characters appearing in the document. Three times are shown: the parse time, the first traversal of the document, and the second traversal of the tree.

This class is useful as a "poor-man's" performance tester to compare the speed and accuracy of various DOM parsers. However, it is important to note that the first parse time of a parser will include both VM class load time and parser initialization that would not be present in subsequent parses with the same file.

The results produced by this program should never be accepted as true performance measurements.

usage

```
java dom.Counter (options) uri ...
```

options

Option	Description
-p name	Select parser wrapper by name.
-x number	Select number of repetitions.
-n | -N	Turn on/off namespace processing.
-v | -V	Turn on/off validation.
-s | -S	Turn on/off Schema validation support. **NOTE:** Not supported by all parsers.
-f | -F	Turn on/off Schema full checking. **NOTE:** Requires use of -s and not supported by all parsers.
-hs | -HS	Turn on/off honouring of all schema locations. **NOTE:** Requires use of -s and not supported by all parsers.
-va | -VA	Turn on/off validation of schema annotations. **NOTE:** Requires use of -s and not supported by all parsers.
-dv | -DV	Turn on/off dynamic validation. **NOTE:** Not supported by all parsers.
-xi | -XI	Turn on/off XInclude processing. **NOTE:** Not supported by all parsers.
-xb | -XB	Turn on/off base URI fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-xl | -XL	Turn on/off language fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-h	Display help screen.

Sample dom.GetElementsByTagName

A sample DOM filter. This sample program illustrates how to use the `Document#getElementsByTagName()` method to quickly and easily locate elements by name.

usage

```
java dom.GetElementsByTagName (options) uri ...
```

options

Option	Description
-p name	Select parser wrapper by name.
-e name	Specify element name for search.
-a name	Specify attribute name for specified elements.
-n | -N	Turn on/off namespace processing.
-v | -V	Turn on/off validation.
-s | -S	Turn on/off Schema validation support. **NOTE:** Not supported by all parsers.
-f | -F	Turn on/off Schema full checking. **NOTE:** Requires use of -s and not supported by all parsers.
-hs | -HS	Turn on/off honouring of all schema locations. **NOTE:** Requires use of -s and not supported by all parsers.

-va | -VA	Turn on/off validation of schema annotations. **NOTE:** Requires use of -s and not supported by all parsers.
-dv | -DV	Turn on/off dynamic validation. **NOTE:** Not supported by all parsers.
-xi | -XI	Turn on/off XInclude processing. **NOTE:** Not supported by all parsers.
-xb | -XB	Turn on/off base URI fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-xl | -XL	Turn on/off language fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-h	Display help screen.

Sample dom.Writer

A sample DOM writer. This sample program illustrates how to traverse a DOM tree in order to print a document that is parsed.

usage

```
java dom.Writer (options) uri ...
```

options

-p name	Select parser wrapper by name.
-n | -N	Turn on/off namespace processing.
-v | -V	Turn on/off validation.
-xd | -XD	Turn on/off loading of external DTDs. **NOTE:** Always on when -v in use and not supported by all parsers.
-s | -S	Turn on/off Schema validation support. **NOTE:** Not supported by all parsers.
-f | -F	Turn on/off Schema full checking. **NOTE:** Requires use of -s and not supported by all parsers.
-hs | -HS	Turn on/off honouring of all schema locations. **NOTE:** Requires use of -s and not supported by all parsers.
-va | -VA	Turn on/off validation of schema annotations. **NOTE:** Requires use of -s and not supported by all parsers.
-ga | -GA	Turn on/off generation of synthetic schema annotations. **NOTE:** Requires use of -s and not supported by all parsers.
-dv | -DV	Turn on/off dynamic validation. **NOTE:** Not supported by all parsers.
-xi | -XI	Turn on/off XInclude processing. **NOTE:** Not supported by all parsers.
-xb | -XB	Turn on/off base URI fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-xl | -XL	Turn on/off language fixup during XInclude processing. **NOTE:** Requires use of -xi and not supported by all parsers.
-c | -C	Turn on/off Canonical XML output. **NOTE:** This is not W3C canonical output.
-h	Display help screen.

Sample dom.ElementPrinter

This sample program illustrates how to use the org.w3c.dom.ElementTraversal API.

usage

```
java dom.ElementPrinter uri
```

Sample dom.DOM3

This sample program illustrates how to use the DOM Level 3 API.

usage

```
java dom.DOM3 uri
```

Sample dom.DOMAddLines

A sample of Adding lines to the DOM Node. This sample program illustrates:

- How to override methods from DocumentHandler (XMLDocumentHandler)
- How to turn off ignorable white spaces by overriding ignorableWhiteSpace
- How to use the SAX Locator to return row position (line number of DOM element)
- How to attach user defined Objects to Nodes using the DOM Level 3 setUserData method.

usage

```
java dom.DOMAddLines (options) uri ...
```

options

Option	Description
-h	Display help screen.
-i	Don't print ignorable white spaces.

Copyright © 1999-2010 The Apache Software Foundation. All Rights Reserved.