| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/LSParser.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/w3c/dom/ls/LSOutput.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/ls/LSParser.html)    [**NO FRAMES**](http://docs.google.com/LSParser.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#3rdcrjn) |

## **org.w3c.dom.ls**

Interface LSParser

public interface **LSParser**

An interface to an object that is able to build, or augment, a DOM tree from various input sources.

LSParser provides an API for parsing XML and building the corresponding DOM document structure. A LSParser instance can be obtained by invoking the DOMImplementationLS.createLSParser() method.

As specified in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , when a document is first made available via the LSParser:

* there will never be two adjacent nodes of type NODE\_TEXT, and there will never be empty text nodes.
* it is expected that the value and nodeValue attributes of an Attr node initially return the [XML 1.0 normalized value](http://www.w3.org/TR/2004/REC-xml-20040204#AVNormalize). However, if the parameters " [validate-if-schema](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-validate-if-schema)" and " [datatype-normalization](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-datatype-normalization)" are set to true, depending on the attribute normalization used, the attribute values may differ from the ones obtained by the XML 1.0 attribute normalization. If the parameters " [datatype-normalization](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-datatype-normalization)" is set to false, the XML 1.0 attribute normalization is guaranteed to occur, and if the attributes list does not contain namespace declarations, the attributes attribute on Element node represents the property **[attributes]** defined in [[XML Information Set](http://www.w3.org/TR/2004/REC-xml-infoset-20040204/)] .

Asynchronous LSParser objects are expected to also implement the events::EventTarget interface so that event listeners can be registered on asynchronous LSParser objects.

Events supported by asynchronous LSParser objects are:

load The LSParser finishes to load the document. See also the definition of the LSLoadEvent interface. progress The LSParser signals progress as data is parsed. This specification does not attempt to define exactly when progress events should be dispatched. That is intentionally left as implementation-dependent. Here is one example of how an application might dispatch progress events: Once the parser starts receiving data, a progress event is dispatched to indicate that the parsing starts. From there on, a progress event is dispatched for every 4096 bytes of data that is received and processed. This is only one example, though, and implementations can choose to dispatch progress events at any time while parsing, or not dispatch them at all. See also the definition of the LSProgressEvent interface.

**Note:** All events defined in this specification use the namespace URI "http://www.w3.org/2002/DOMLS".

While parsing an input source, errors are reported to the application through the error handler (LSParser.domConfig's " [error-handler](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-error-handler)" parameter). This specification does in no way try to define all possible errors that can occur while parsing XML, or any other markup, but some common error cases are defined. The types (DOMError.type) of errors and warnings defined by this specification are:

"check-character-normalization-failure" [error] Raised if the parameter " [check-character-normalization](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-check-character-normalization)" is set to true and a string is encountered that fails normalization checking. "doctype-not-allowed" [fatal] Raised if the configuration parameter "disallow-doctype" is set to true and a doctype is encountered. "no-input-specified" [fatal] Raised when loading a document and no input is specified in the LSInput object. "pi-base-uri-not-preserved" [warning] Raised if a processing instruction is encountered in a location where the base URI of the processing instruction can not be preserved. One example of a case where this warning will be raised is if the configuration parameter " [entities](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-entities)" is set to false and the following XML file is parsed:

<!DOCTYPE root [ <!ENTITY e SYSTEM 'subdir/myentity.ent' ]>   
 <root> &e; </root>

And subdir/myentity.ent contains:

<one> <two/> </one> <?pi 3.14159?>   
 <more/>

"unbound-prefix-in-entity" [warning] An implementation dependent warning that may be raised if the configuration parameter " [namespaces](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-namespaces)" is set to true and an unbound namespace prefix is encountered in an entity's replacement text. Raising this warning is not enforced since some existing parsers may not recognize unbound namespace prefixes in the replacement text of entities. "unknown-character-denormalization" [fatal] Raised if the configuration parameter "ignore-unknown-character-denormalizations" is set to false and a character is encountered for which the processor cannot determine the normalization properties. "unsupported-encoding" [fatal] Raised if an unsupported encoding is encountered. "unsupported-media-type" [fatal] Raised if the configuration parameter "supported-media-types-only" is set to true and an unsupported media type is encountered.

In addition to raising the defined errors and warnings, implementations are expected to raise implementation specific errors and warnings for any other error and warning cases such as IO errors (file not found, permission denied,...), XML well-formedness errors, and so on.

See also the [Document Object Model (DOM) Level 3 Load and Save Specification](http://www.w3.org/TR/2004/REC-DOM-Level-3-LS-20040407).

| **Field Summary** | |
| --- | --- |
| static short | [**ACTION\_APPEND\_AS\_CHILDREN**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#ACTION_APPEND_AS_CHILDREN)            Append the result of the parse operation as children of the context node. |
| static short | [**ACTION\_INSERT\_AFTER**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#ACTION_INSERT_AFTER)            Insert the result of the parse operation as the immediately following sibling of the context node. |
| static short | [**ACTION\_INSERT\_BEFORE**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#ACTION_INSERT_BEFORE)            Insert the result of the parse operation as the immediately preceding sibling of the context node. |
| static short | [**ACTION\_REPLACE**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#ACTION_REPLACE)            Replace the context node with the result of the parse operation. |
| static short | [**ACTION\_REPLACE\_CHILDREN**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#ACTION_REPLACE_CHILDREN)            Replace all the children of the context node with the result of the parse operation. |

| **Method Summary** | |
| --- | --- |
| void | [**abort**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#abort())()            Abort the loading of the document that is currently being loaded by the LSParser. |
| boolean | [**getAsync**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#getAsync())()            true if the LSParser is asynchronous, false if it is synchronous. |
| boolean | [**getBusy**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#getBusy())()            true if the LSParser is currently busy loading a document, otherwise false. |
| [DOMConfiguration](http://docs.google.com/org/w3c/dom/DOMConfiguration.html) | [**getDomConfig**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#getDomConfig())()            The DOMConfiguration object used when parsing an input source. |
| [LSParserFilter](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) | [**getFilter**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#getFilter())()            When a filter is provided, the implementation will call out to the filter as it is constructing the DOM tree structure. |
| [Document](http://docs.google.com/org/w3c/dom/Document.html) | [**parse**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#parse(org.w3c.dom.ls.LSInput))([LSInput](http://docs.google.com/org/w3c/dom/ls/LSInput.html) input)            Parse an XML document from a resource identified by a LSInput. |
| [Document](http://docs.google.com/org/w3c/dom/Document.html) | [**parseURI**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#parseURI(java.lang.String))([String](http://docs.google.com/java/lang/String.html) uri)            Parse an XML document from a location identified by a URI reference [[IETF RFC 2396](http://www.ietf.org/rfc/rfc2396.txt)]. |
| [Node](http://docs.google.com/org/w3c/dom/Node.html) | [**parseWithContext**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#parseWithContext(org.w3c.dom.ls.LSInput,%20org.w3c.dom.Node,%20short))([LSInput](http://docs.google.com/org/w3c/dom/ls/LSInput.html) input, [Node](http://docs.google.com/org/w3c/dom/Node.html) contextArg, short action)            Parse an XML fragment from a resource identified by a LSInput and insert the content into an existing document at the position specified with the context and action arguments. |
| void | [**setFilter**](http://docs.google.com/org/w3c/dom/ls/LSParser.html#setFilter(org.w3c.dom.ls.LSParserFilter))([LSParserFilter](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) filter)            When a filter is provided, the implementation will call out to the filter as it is constructing the DOM tree structure. |

| **Field Detail** |
| --- |

### ACTION\_APPEND\_AS\_CHILDREN

static final short **ACTION\_APPEND\_AS\_CHILDREN**

Append the result of the parse operation as children of the context node. For this action to work, the context node must be an Element or a DocumentFragment.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.ls.LSParser.ACTION_APPEND_AS_CHILDREN)

### ACTION\_REPLACE\_CHILDREN

static final short **ACTION\_REPLACE\_CHILDREN**

Replace all the children of the context node with the result of the parse operation. For this action to work, the context node must be an Element, a Document, or a DocumentFragment.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.ls.LSParser.ACTION_REPLACE_CHILDREN)

### ACTION\_INSERT\_BEFORE

static final short **ACTION\_INSERT\_BEFORE**

Insert the result of the parse operation as the immediately preceding sibling of the context node. For this action to work the context node's parent must be an Element or a DocumentFragment.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.ls.LSParser.ACTION_INSERT_BEFORE)

### ACTION\_INSERT\_AFTER

static final short **ACTION\_INSERT\_AFTER**

Insert the result of the parse operation as the immediately following sibling of the context node. For this action to work the context node's parent must be an Element or a DocumentFragment.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.ls.LSParser.ACTION_INSERT_AFTER)

### ACTION\_REPLACE

static final short **ACTION\_REPLACE**

Replace the context node with the result of the parse operation. For this action to work, the context node must have a parent, and the parent must be an Element or a DocumentFragment.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.ls.LSParser.ACTION_REPLACE)

| **Method Detail** |
| --- |

### getDomConfig

[DOMConfiguration](http://docs.google.com/org/w3c/dom/DOMConfiguration.html) **getDomConfig**()

The DOMConfiguration object used when parsing an input source. This DOMConfiguration is specific to the parse operation. No parameter values from this DOMConfiguration object are passed automatically to the DOMConfiguration object on the Document that is created, or used, by the parse operation. The DOM application is responsible for passing any needed parameter values from this DOMConfiguration object to the DOMConfiguration object referenced by the Document object.

In addition to the parameters recognized in on the  [DOMConfiguration](http://www.w3.org/TR/DOM-Level-3-Core/core.html#DOMConfiguration) interface defined in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , the DOMConfiguration objects for LSParser add or modify the following parameters: "charset-overrides-xml-encoding" true [*optional*] (*default*) If a higher level protocol such as HTTP [[IETF RFC 2616](http://www.ietf.org/rfc/rfc2616.txt)] provides an indication of the character encoding of the input stream being processed, that will override any encoding specified in the XML declaration or the Text declaration (see also section 4.3.3, "Character Encoding in Entities", in [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)]). Explicitly setting an encoding in the LSInput overrides any encoding from the protocol. false [*required*] The parser ignores any character set encoding information from higher-level protocols. "disallow-doctype" true [*optional*] Throw a fatal **"doctype-not-allowed"** error if a doctype node is found while parsing the document. This is useful when dealing with things like SOAP envelopes where doctype nodes are not allowed. false [*required*] (*default*) Allow doctype nodes in the document. "ignore-unknown-character-denormalizations" true [*required*] (*default*) If, while verifying full normalization when [[XML 1.1](http://www.w3.org/TR/2004/REC-xml11-20040204/)] is supported, a processor encounters characters for which it cannot determine the normalization properties, then the processor will ignore any possible denormalizations caused by these characters. This parameter is ignored for [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)]. false [*optional*] Report an fatal **"unknown-character-denormalization"** error if a character is encountered for which the processor cannot determine the normalization properties. "infoset" See the definition of DOMConfiguration for a description of this parameter. Unlike in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , this parameter will default to true for LSParser. "namespaces" true [*required*] (*default*) Perform the namespace processing as defined in [[XML Namespaces](http://www.w3.org/TR/1999/REC-xml-names-19990114/)] and [[XML Namespaces 1.1](http://www.w3.org/TR/2004/REC-xml-names11-20040204/)] . false [*optional*] Do not perform the namespace processing. "resource-resolver" [*required*] A reference to a LSResourceResolver object, or null. If the value of this parameter is not null when an external resource (such as an external XML entity or an XML schema location) is encountered, the implementation will request that the LSResourceResolver referenced in this parameter resolves the resource. "supported-media-types-only" true [*optional*] Check that the media type of the parsed resource is a supported media type. If an unsupported media type is encountered, a fatal error of type **"unsupported-media-type"** will be raised. The media types defined in [[IETF RFC 3023](http://www.ietf.org/rfc/rfc3023.txt)] must always be accepted. false [*required*] (*default*) Accept any media type. "validate" See the definition of DOMConfiguration for a description of this parameter. Unlike in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , the processing of the internal subset is always accomplished, even if this parameter is set to false. "validate-if-schema" See the definition of DOMConfiguration for a description of this parameter. Unlike in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , the processing of the internal subset is always accomplished, even if this parameter is set to false. "well-formed" See the definition of DOMConfiguration for a description of this parameter. Unlike in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] , this parameter cannot be set to false.

### getFilter

[LSParserFilter](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) **getFilter**()

When a filter is provided, the implementation will call out to the filter as it is constructing the DOM tree structure. The filter can choose to remove elements from the document being constructed, or to terminate the parsing early.

The filter is invoked after the operations requested by the DOMConfiguration parameters have been applied. For example, if " [validate](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-validate)" is set to true, the validation is done before invoking the filter.

### setFilter

void **setFilter**([LSParserFilter](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) filter)

When a filter is provided, the implementation will call out to the filter as it is constructing the DOM tree structure. The filter can choose to remove elements from the document being constructed, or to terminate the parsing early.

The filter is invoked after the operations requested by the DOMConfiguration parameters have been applied. For example, if " [validate](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-validate)" is set to true, the validation is done before invoking the filter.

### getAsync

boolean **getAsync**()

true if the LSParser is asynchronous, false if it is synchronous.

### getBusy

boolean **getBusy**()

true if the LSParser is currently busy loading a document, otherwise false.

### parse

[Document](http://docs.google.com/org/w3c/dom/Document.html) **parse**([LSInput](http://docs.google.com/org/w3c/dom/ls/LSInput.html) input)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html),  
 [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html)

Parse an XML document from a resource identified by a LSInput.

**Parameters:**input - The LSInput from which the source of the document is to be read. **Returns:**If the LSParser is a synchronous LSParser, the newly created and populated Document is returned. If the LSParser is asynchronous, null is returned since the document object may not yet be constructed when this method returns. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_STATE\_ERR: Raised if the LSParser's LSParser.busy attribute is true. [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html) - PARSE\_ERR: Raised if the LSParser was unable to load the XML document. DOM applications should attach a DOMErrorHandler using the parameter " [error-handler](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-error-handler)" if they wish to get details on the error.

### parseURI

[Document](http://docs.google.com/org/w3c/dom/Document.html) **parseURI**([String](http://docs.google.com/java/lang/String.html) uri)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html),  
 [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html)

Parse an XML document from a location identified by a URI reference [[IETF RFC 2396](http://www.ietf.org/rfc/rfc2396.txt)]. If the URI contains a fragment identifier (see section 4.1 in [[IETF RFC 2396](http://www.ietf.org/rfc/rfc2396.txt)]), the behavior is not defined by this specification, future versions of this specification may define the behavior.

**Parameters:**uri - The location of the XML document to be read. **Returns:**If the LSParser is a synchronous LSParser, the newly created and populated Document is returned, or null if an error occured. If the LSParser is asynchronous, null is returned since the document object may not yet be constructed when this method returns. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_STATE\_ERR: Raised if the LSParser.busy attribute is true. [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html) - PARSE\_ERR: Raised if the LSParser was unable to load the XML document. DOM applications should attach a DOMErrorHandler using the parameter " [error-handler](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-error-handler)" if they wish to get details on the error.

### parseWithContext

[Node](http://docs.google.com/org/w3c/dom/Node.html) **parseWithContext**([LSInput](http://docs.google.com/org/w3c/dom/ls/LSInput.html) input,  
 [Node](http://docs.google.com/org/w3c/dom/Node.html) contextArg,  
 short action)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html),  
 [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html)

Parse an XML fragment from a resource identified by a LSInput and insert the content into an existing document at the position specified with the context and action arguments. When parsing the input stream, the context node (or its parent, depending on where the result will be inserted) is used for resolving unbound namespace prefixes. The context node's ownerDocument node (or the node itself if the node of type DOCUMENT\_NODE) is used to resolve default attributes and entity references.

As the new data is inserted into the document, at least one mutation event is fired per new immediate child or sibling of the context node.

If the context node is a Document node and the action is ACTION\_REPLACE\_CHILDREN, then the document that is passed as the context node will be changed such that its xmlEncoding, documentURI, xmlVersion, inputEncoding, xmlStandalone, and all other such attributes are set to what they would be set to if the input source was parsed using LSParser.parse().

This method is always synchronous, even if the LSParser is asynchronous (LSParser.async is true).

If an error occurs while parsing, the caller is notified through the ErrorHandler instance associated with the " [error-handler](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-error-handler)" parameter of the DOMConfiguration.

When calling parseWithContext, the values of the following configuration parameters will be ignored and their default values will always be used instead: " [validate](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-validate)", " [validate-if-schema](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-validate-if-schema)", and " [element-content-whitespace](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-element-content-whitespace)". Other parameters will be treated normally, and the parser is expected to call the LSParserFilter just as if a whole document was parsed.

**Parameters:**input - The LSInput from which the source document is to be read. The source document must be an XML fragment, i.e. anything except a complete XML document (except in the case where the context node of type DOCUMENT\_NODE, and the action is ACTION\_REPLACE\_CHILDREN), a DOCTYPE (internal subset), entity declaration(s), notation declaration(s), or XML or text declaration(s).contextArg - The node that is used as the context for the data that is being parsed. This node must be a Document node, a DocumentFragment node, or a node of a type that is allowed as a child of an Element node, e.g. it cannot be an Attribute node.action - This parameter describes which action should be taken between the new set of nodes being inserted and the existing children of the context node. The set of possible actions is defined in ACTION\_TYPES above. **Returns:**Return the node that is the result of the parse operation. If the result is more than one top-level node, the first one is returned. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - HIERARCHY\_REQUEST\_ERR: Raised if the content cannot replace, be inserted before, after, or as a child of the context node (see also Node.insertBefore or Node.replaceChild in [[DOM Level 3 Core](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407)] ).

NOT\_SUPPORTED\_ERR: Raised if the LSParser doesn't support this method, or if the context node is of type Document and the DOM implementation doesn't support the replacement of the DocumentType child or Element child.

NO\_MODIFICATION\_ALLOWED\_ERR: Raised if the context node is a read only node and the content is being appended to its child list, or if the parent node of the context node is read only node and the content is being inserted in its child list.

INVALID\_STATE\_ERR: Raised if the LSParser.busy attribute is true. [LSException](http://docs.google.com/org/w3c/dom/ls/LSException.html) - PARSE\_ERR: Raised if the LSParser was unable to load the XML fragment. DOM applications should attach a DOMErrorHandler using the parameter " [error-handler](http://www.w3.org/TR/DOM-Level-3-Core/core.html#parameter-error-handler)" if they wish to get details on the error.

### abort

void **abort**()

Abort the loading of the document that is currently being loaded by the LSParser. If the LSParser is currently not busy, a call to this method does nothing.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/LSParser.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/w3c/dom/ls/LSOutput.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/ls/LSParserFilter.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/ls/LSParser.html)    [**NO FRAMES**](http://docs.google.com/LSParser.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#3rdcrjn) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).