| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Document.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/Comment.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/DocumentFragment.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/Document.html)    [**NO FRAMES**](http://docs.google.com/Document.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#1t3h5sf) |

## **org.w3c.dom**

Interface Document

**All Superinterfaces:** [Node](http://docs.google.com/org/w3c/dom/Node.html) **All Known Implementing Classes:** [SOAPPart](http://docs.google.com/javax/xml/soap/SOAPPart.html)

public interface **Document**extends [Node](http://docs.google.com/org/w3c/dom/Node.html)

The Document interface represents the entire HTML or XML document. Conceptually, it is the root of the document tree, and provides the primary access to the document's data.

Since elements, text nodes, comments, processing instructions, etc. cannot exist outside the context of a Document, the Document interface also contains the factory methods needed to create these objects. The Node objects created have a ownerDocument attribute which associates them with the Document within whose context they were created.

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

| **Field Summary** | |
| --- | --- |

| **Fields inherited from interface org.w3c.dom.**[**Node**](http://docs.google.com/org/w3c/dom/Node.html) |
| --- |
| [ATTRIBUTE\_NODE](http://docs.google.com/org/w3c/dom/Node.html#ATTRIBUTE_NODE), [CDATA\_SECTION\_NODE](http://docs.google.com/org/w3c/dom/Node.html#CDATA_SECTION_NODE), [COMMENT\_NODE](http://docs.google.com/org/w3c/dom/Node.html#COMMENT_NODE), [DOCUMENT\_FRAGMENT\_NODE](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_FRAGMENT_NODE), [DOCUMENT\_NODE](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_NODE), [DOCUMENT\_POSITION\_CONTAINED\_BY](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_CONTAINED_BY), [DOCUMENT\_POSITION\_CONTAINS](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_CONTAINS), [DOCUMENT\_POSITION\_DISCONNECTED](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_DISCONNECTED), [DOCUMENT\_POSITION\_FOLLOWING](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_FOLLOWING), [DOCUMENT\_POSITION\_IMPLEMENTATION\_SPECIFIC](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_IMPLEMENTATION_SPECIFIC), [DOCUMENT\_POSITION\_PRECEDING](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_POSITION_PRECEDING), [DOCUMENT\_TYPE\_NODE](http://docs.google.com/org/w3c/dom/Node.html#DOCUMENT_TYPE_NODE), [ELEMENT\_NODE](http://docs.google.com/org/w3c/dom/Node.html#ELEMENT_NODE), [ENTITY\_NODE](http://docs.google.com/org/w3c/dom/Node.html#ENTITY_NODE), [ENTITY\_REFERENCE\_NODE](http://docs.google.com/org/w3c/dom/Node.html#ENTITY_REFERENCE_NODE), [NOTATION\_NODE](http://docs.google.com/org/w3c/dom/Node.html#NOTATION_NODE), [PROCESSING\_INSTRUCTION\_NODE](http://docs.google.com/org/w3c/dom/Node.html#PROCESSING_INSTRUCTION_NODE), [TEXT\_NODE](http://docs.google.com/org/w3c/dom/Node.html#TEXT_NODE) |

| **Method Summary** | |
| --- | --- |
| [Node](http://docs.google.com/org/w3c/dom/Node.html) | [**adoptNode**](http://docs.google.com/org/w3c/dom/Document.html#adoptNode(org.w3c.dom.Node))([Node](http://docs.google.com/org/w3c/dom/Node.html) source)            Attempts to adopt a node from another document to this document. |
| [Attr](http://docs.google.com/org/w3c/dom/Attr.html) | [**createAttribute**](http://docs.google.com/org/w3c/dom/Document.html#createAttribute(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Creates an Attr of the given name. |
| [Attr](http://docs.google.com/org/w3c/dom/Attr.html) | [**createAttributeNS**](http://docs.google.com/org/w3c/dom/Document.html#createAttributeNS(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) namespaceURI, [String](http://docs.google.com/java/lang/String.html) qualifiedName)            Creates an attribute of the given qualified name and namespace URI. |
| [CDATASection](http://docs.google.com/org/w3c/dom/CDATASection.html) | [**createCDATASection**](http://docs.google.com/org/w3c/dom/Document.html#createCDATASection(java.lang.String))([String](http://docs.google.com/java/lang/String.html) data)            Creates a CDATASection node whose value is the specified string. |
| [Comment](http://docs.google.com/org/w3c/dom/Comment.html) | [**createComment**](http://docs.google.com/org/w3c/dom/Document.html#createComment(java.lang.String))([String](http://docs.google.com/java/lang/String.html) data)            Creates a Comment node given the specified string. |
| [DocumentFragment](http://docs.google.com/org/w3c/dom/DocumentFragment.html) | [**createDocumentFragment**](http://docs.google.com/org/w3c/dom/Document.html#createDocumentFragment())()            Creates an empty DocumentFragment object. |
| [Element](http://docs.google.com/org/w3c/dom/Element.html) | [**createElement**](http://docs.google.com/org/w3c/dom/Document.html#createElement(java.lang.String))([String](http://docs.google.com/java/lang/String.html) tagName)            Creates an element of the type specified. |
| [Element](http://docs.google.com/org/w3c/dom/Element.html) | [**createElementNS**](http://docs.google.com/org/w3c/dom/Document.html#createElementNS(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) namespaceURI, [String](http://docs.google.com/java/lang/String.html) qualifiedName)            Creates an element of the given qualified name and namespace URI. |
| [EntityReference](http://docs.google.com/org/w3c/dom/EntityReference.html) | [**createEntityReference**](http://docs.google.com/org/w3c/dom/Document.html#createEntityReference(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Creates an EntityReference object. |
| [ProcessingInstruction](http://docs.google.com/org/w3c/dom/ProcessingInstruction.html) | [**createProcessingInstruction**](http://docs.google.com/org/w3c/dom/Document.html#createProcessingInstruction(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) target, [String](http://docs.google.com/java/lang/String.html) data)            Creates a ProcessingInstruction node given the specified name and data strings. |
| [Text](http://docs.google.com/org/w3c/dom/Text.html) | [**createTextNode**](http://docs.google.com/org/w3c/dom/Document.html#createTextNode(java.lang.String))([String](http://docs.google.com/java/lang/String.html) data)            Creates a Text node given the specified string. |
| [DocumentType](http://docs.google.com/org/w3c/dom/DocumentType.html) | [**getDoctype**](http://docs.google.com/org/w3c/dom/Document.html#getDoctype())()            The Document Type Declaration (see DocumentType) associated with this document. |
| [Element](http://docs.google.com/org/w3c/dom/Element.html) | [**getDocumentElement**](http://docs.google.com/org/w3c/dom/Document.html#getDocumentElement())()            This is a convenience attribute that allows direct access to the child node that is the document element of the document. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDocumentURI**](http://docs.google.com/org/w3c/dom/Document.html#getDocumentURI())()            The location of the document or null if undefined or if the Document was created using DOMImplementation.createDocument. |
| [DOMConfiguration](http://docs.google.com/org/w3c/dom/DOMConfiguration.html) | [**getDomConfig**](http://docs.google.com/org/w3c/dom/Document.html#getDomConfig())()            The configuration used when Document.normalizeDocument() is invoked. |
| [Element](http://docs.google.com/org/w3c/dom/Element.html) | [**getElementById**](http://docs.google.com/org/w3c/dom/Document.html#getElementById(java.lang.String))([String](http://docs.google.com/java/lang/String.html) elementId)            Returns the Element that has an ID attribute with the given value. |
| [NodeList](http://docs.google.com/org/w3c/dom/NodeList.html) | [**getElementsByTagName**](http://docs.google.com/org/w3c/dom/Document.html#getElementsByTagName(java.lang.String))([String](http://docs.google.com/java/lang/String.html) tagname)            Returns a NodeList of all the Elements in document order with a given tag name and are contained in the document. |
| [NodeList](http://docs.google.com/org/w3c/dom/NodeList.html) | [**getElementsByTagNameNS**](http://docs.google.com/org/w3c/dom/Document.html#getElementsByTagNameNS(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) namespaceURI, [String](http://docs.google.com/java/lang/String.html) localName)            Returns a NodeList of all the Elements with a given local name and namespace URI in document order. |
| [DOMImplementation](http://docs.google.com/org/w3c/dom/DOMImplementation.html) | [**getImplementation**](http://docs.google.com/org/w3c/dom/Document.html#getImplementation())()            The DOMImplementation object that handles this document. |
| [String](http://docs.google.com/java/lang/String.html) | [**getInputEncoding**](http://docs.google.com/org/w3c/dom/Document.html#getInputEncoding())()            An attribute specifying the encoding used for this document at the time of the parsing. |
| boolean | [**getStrictErrorChecking**](http://docs.google.com/org/w3c/dom/Document.html#getStrictErrorChecking())()            An attribute specifying whether error checking is enforced or not. |
| [String](http://docs.google.com/java/lang/String.html) | [**getXmlEncoding**](http://docs.google.com/org/w3c/dom/Document.html#getXmlEncoding())()            An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the encoding of this document. |
| boolean | [**getXmlStandalone**](http://docs.google.com/org/w3c/dom/Document.html#getXmlStandalone())()            An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), whether this document is standalone. |
| [String](http://docs.google.com/java/lang/String.html) | [**getXmlVersion**](http://docs.google.com/org/w3c/dom/Document.html#getXmlVersion())()            An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the version number of this document. |
| [Node](http://docs.google.com/org/w3c/dom/Node.html) | [**importNode**](http://docs.google.com/org/w3c/dom/Document.html#importNode(org.w3c.dom.Node,%20boolean))([Node](http://docs.google.com/org/w3c/dom/Node.html) importedNode, boolean deep)            Imports a node from another document to this document, without altering or removing the source node from the original document; this method creates a new copy of the source node. |
| void | [**normalizeDocument**](http://docs.google.com/org/w3c/dom/Document.html#normalizeDocument())()            This method acts as if the document was going through a save and load cycle, putting the document in a "normal" form. |
| [Node](http://docs.google.com/org/w3c/dom/Node.html) | [**renameNode**](http://docs.google.com/org/w3c/dom/Document.html#renameNode(org.w3c.dom.Node,%20java.lang.String,%20java.lang.String))([Node](http://docs.google.com/org/w3c/dom/Node.html) n, [String](http://docs.google.com/java/lang/String.html) namespaceURI, [String](http://docs.google.com/java/lang/String.html) qualifiedName)            Rename an existing node of type ELEMENT\_NODE or ATTRIBUTE\_NODE. |
| void | [**setDocumentURI**](http://docs.google.com/org/w3c/dom/Document.html#setDocumentURI(java.lang.String))([String](http://docs.google.com/java/lang/String.html) documentURI)            The location of the document or null if undefined or if the Document was created using DOMImplementation.createDocument. |
| void | [**setStrictErrorChecking**](http://docs.google.com/org/w3c/dom/Document.html#setStrictErrorChecking(boolean))(boolean strictErrorChecking)            An attribute specifying whether error checking is enforced or not. |
| void | [**setXmlStandalone**](http://docs.google.com/org/w3c/dom/Document.html#setXmlStandalone(boolean))(boolean xmlStandalone)            An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), whether this document is standalone. |
| void | [**setXmlVersion**](http://docs.google.com/org/w3c/dom/Document.html#setXmlVersion(java.lang.String))([String](http://docs.google.com/java/lang/String.html) xmlVersion)            An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the version number of this document. |

| **Methods inherited from interface org.w3c.dom.**[**Node**](http://docs.google.com/org/w3c/dom/Node.html) |
| --- |
| [appendChild](http://docs.google.com/org/w3c/dom/Node.html#appendChild(org.w3c.dom.Node)), [cloneNode](http://docs.google.com/org/w3c/dom/Node.html#cloneNode(boolean)), [compareDocumentPosition](http://docs.google.com/org/w3c/dom/Node.html#compareDocumentPosition(org.w3c.dom.Node)), [getAttributes](http://docs.google.com/org/w3c/dom/Node.html#getAttributes()), [getBaseURI](http://docs.google.com/org/w3c/dom/Node.html#getBaseURI()), [getChildNodes](http://docs.google.com/org/w3c/dom/Node.html#getChildNodes()), [getFeature](http://docs.google.com/org/w3c/dom/Node.html#getFeature(java.lang.String,%20java.lang.String)), [getFirstChild](http://docs.google.com/org/w3c/dom/Node.html#getFirstChild()), [getLastChild](http://docs.google.com/org/w3c/dom/Node.html#getLastChild()), [getLocalName](http://docs.google.com/org/w3c/dom/Node.html#getLocalName()), [getNamespaceURI](http://docs.google.com/org/w3c/dom/Node.html#getNamespaceURI()), [getNextSibling](http://docs.google.com/org/w3c/dom/Node.html#getNextSibling()), [getNodeName](http://docs.google.com/org/w3c/dom/Node.html#getNodeName()), [getNodeType](http://docs.google.com/org/w3c/dom/Node.html#getNodeType()), [getNodeValue](http://docs.google.com/org/w3c/dom/Node.html#getNodeValue()), [getOwnerDocument](http://docs.google.com/org/w3c/dom/Node.html#getOwnerDocument()), [getParentNode](http://docs.google.com/org/w3c/dom/Node.html#getParentNode()), [getPrefix](http://docs.google.com/org/w3c/dom/Node.html#getPrefix()), [getPreviousSibling](http://docs.google.com/org/w3c/dom/Node.html#getPreviousSibling()), [getTextContent](http://docs.google.com/org/w3c/dom/Node.html#getTextContent()), [getUserData](http://docs.google.com/org/w3c/dom/Node.html#getUserData(java.lang.String)), [hasAttributes](http://docs.google.com/org/w3c/dom/Node.html#hasAttributes()), [hasChildNodes](http://docs.google.com/org/w3c/dom/Node.html#hasChildNodes()), [insertBefore](http://docs.google.com/org/w3c/dom/Node.html#insertBefore(org.w3c.dom.Node,%20org.w3c.dom.Node)), [isDefaultNamespace](http://docs.google.com/org/w3c/dom/Node.html#isDefaultNamespace(java.lang.String)), [isEqualNode](http://docs.google.com/org/w3c/dom/Node.html#isEqualNode(org.w3c.dom.Node)), [isSameNode](http://docs.google.com/org/w3c/dom/Node.html#isSameNode(org.w3c.dom.Node)), [isSupported](http://docs.google.com/org/w3c/dom/Node.html#isSupported(java.lang.String,%20java.lang.String)), [lookupNamespaceURI](http://docs.google.com/org/w3c/dom/Node.html#lookupNamespaceURI(java.lang.String)), [lookupPrefix](http://docs.google.com/org/w3c/dom/Node.html#lookupPrefix(java.lang.String)), [normalize](http://docs.google.com/org/w3c/dom/Node.html#normalize()), [removeChild](http://docs.google.com/org/w3c/dom/Node.html#removeChild(org.w3c.dom.Node)), [replaceChild](http://docs.google.com/org/w3c/dom/Node.html#replaceChild(org.w3c.dom.Node,%20org.w3c.dom.Node)), [setNodeValue](http://docs.google.com/org/w3c/dom/Node.html#setNodeValue(java.lang.String)), [setPrefix](http://docs.google.com/org/w3c/dom/Node.html#setPrefix(java.lang.String)), [setTextContent](http://docs.google.com/org/w3c/dom/Node.html#setTextContent(java.lang.String)), [setUserData](http://docs.google.com/org/w3c/dom/Node.html#setUserData(java.lang.String,%20java.lang.Object,%20org.w3c.dom.UserDataHandler)) |

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

### getDoctype

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

The Document Type Declaration (see DocumentType) associated with this document. For XML documents without a document type declaration this returns null. For HTML documents, a DocumentType object may be returned, independently of the presence or absence of document type declaration in the HTML document.

This provides direct access to the DocumentType node, child node of this Document. This node can be set at document creation time and later changed through the use of child nodes manipulation methods, such as Node.insertBefore, or Node.replaceChild. Note, however, that while some implementations may instantiate different types of Document objects supporting additional features than the "Core", such as "HTML" [[DOM Level 2 HTML](http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109)] , based on the DocumentType specified at creation time, changing it afterwards is very unlikely to result in a change of the features supported.

**Since:** DOM Level 3

### getImplementation

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

The DOMImplementation object that handles this document. A DOM application may use objects from multiple implementations.

### getDocumentElement

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

This is a convenience attribute that allows direct access to the child node that is the document element of the document.

### createElement

[Element](http://docs.google.com/org/w3c/dom/Element.html) **createElement**([String](http://docs.google.com/java/lang/String.html) tagName)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates an element of the type specified. Note that the instance returned implements the Element interface, so attributes can be specified directly on the returned object.

In addition, if there are known attributes with default values, Attr nodes representing them are automatically created and attached to the element.

To create an element with a qualified name and namespace URI, use the createElementNS method.

**Parameters:**tagName - The name of the element type to instantiate. For XML, this is case-sensitive, otherwise it depends on the case-sensitivity of the markup language in use. In that case, the name is mapped to the canonical form of that markup by the DOM implementation. **Returns:**A new Element object with the nodeName attribute set to tagName, and localName, prefix, and namespaceURI set to null. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified name is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

### createDocumentFragment

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

Creates an empty DocumentFragment object.

**Returns:**A new DocumentFragment.

### createTextNode

[Text](http://docs.google.com/org/w3c/dom/Text.html) **createTextNode**([String](http://docs.google.com/java/lang/String.html) data)

Creates a Text node given the specified string.

**Parameters:**data - The data for the node. **Returns:**The new Text object.

### createComment

[Comment](http://docs.google.com/org/w3c/dom/Comment.html) **createComment**([String](http://docs.google.com/java/lang/String.html) data)

Creates a Comment node given the specified string.

**Parameters:**data - The data for the node. **Returns:**The new Comment object.

### createCDATASection

[CDATASection](http://docs.google.com/org/w3c/dom/CDATASection.html) **createCDATASection**([String](http://docs.google.com/java/lang/String.html) data)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates a CDATASection node whose value is the specified string.

**Parameters:**data - The data for the CDATASection contents. **Returns:**The new CDATASection object. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised if this document is an HTML document.

### createProcessingInstruction

[ProcessingInstruction](http://docs.google.com/org/w3c/dom/ProcessingInstruction.html) **createProcessingInstruction**([String](http://docs.google.com/java/lang/String.html) target,  
 [String](http://docs.google.com/java/lang/String.html) data)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates a ProcessingInstruction node given the specified name and data strings.

**Parameters:**target - The target part of the processing instruction.Unlike Document.createElementNS or Document.createAttributeNS, no namespace well-formed checking is done on the target name. Applications should invoke Document.normalizeDocument() with the parameter " namespaces" set to true in order to ensure that the target name is namespace well-formed.data - The data for the node. **Returns:**The new ProcessingInstruction object. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified target is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

NOT\_SUPPORTED\_ERR: Raised if this document is an HTML document.

### createAttribute

[Attr](http://docs.google.com/org/w3c/dom/Attr.html) **createAttribute**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates an Attr of the given name. Note that the Attr instance can then be set on an Element using the setAttributeNode method.

To create an attribute with a qualified name and namespace URI, use the createAttributeNS method.

**Parameters:**name - The name of the attribute. **Returns:**A new Attr object with the nodeName attribute set to name, and localName, prefix, and namespaceURI set to null. The value of the attribute is the empty string. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified name is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

### createEntityReference

[EntityReference](http://docs.google.com/org/w3c/dom/EntityReference.html) **createEntityReference**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates an EntityReference object. In addition, if the referenced entity is known, the child list of the EntityReference node is made the same as that of the corresponding Entity node.

**Note:** If any descendant of the Entity node has an unbound namespace prefix, the corresponding descendant of the created EntityReference node is also unbound; (its namespaceURI is null). The DOM Level 2 and 3 do not support any mechanism to resolve namespace prefixes in this case.

**Parameters:**name - The name of the entity to reference.Unlike Document.createElementNS or Document.createAttributeNS, no namespace well-formed checking is done on the entity name. Applications should invoke Document.normalizeDocument() with the parameter " namespaces" set to true in order to ensure that the entity name is namespace well-formed. **Returns:**The new EntityReference object. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified name is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

NOT\_SUPPORTED\_ERR: Raised if this document is an HTML document.

### getElementsByTagName

[NodeList](http://docs.google.com/org/w3c/dom/NodeList.html) **getElementsByTagName**([String](http://docs.google.com/java/lang/String.html) tagname)

Returns a NodeList of all the Elements in document order with a given tag name and are contained in the document.

**Parameters:**tagname - The name of the tag to match on. The special value "\*" matches all tags. For XML, the tagname parameter is case-sensitive, otherwise it depends on the case-sensitivity of the markup language in use. **Returns:**A new NodeList object containing all the matched Elements.

### importNode

[Node](http://docs.google.com/org/w3c/dom/Node.html) **importNode**([Node](http://docs.google.com/org/w3c/dom/Node.html) importedNode,  
 boolean deep)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Imports a node from another document to this document, without altering or removing the source node from the original document; this method creates a new copy of the source node. The returned node has no parent; (parentNode is null).

For all nodes, importing a node creates a node object owned by the importing document, with attribute values identical to the source node's nodeName and nodeType, plus the attributes related to namespaces (prefix, localName, and namespaceURI). As in the cloneNode operation, the source node is not altered. User data associated to the imported node is not carried over. However, if any UserDataHandlers has been specified along with the associated data these handlers will be called with the appropriate parameters before this method returns.

Additional information is copied as appropriate to the nodeType, attempting to mirror the behavior expected if a fragment of XML or HTML source was copied from one document to another, recognizing that the two documents may have different DTDs in the XML case. The following list describes the specifics for each type of node. ATTRIBUTE\_NODE The ownerElement attribute is set to null and the specified flag is set to true on the generated Attr. The descendants of the source Attr are recursively imported and the resulting nodes reassembled to form the corresponding subtree. Note that the deep parameter has no effect on Attr nodes; they always carry their children with them when imported. DOCUMENT\_FRAGMENT\_NODE If the deep option was set to true, the descendants of the source DocumentFragment are recursively imported and the resulting nodes reassembled under the imported DocumentFragment to form the corresponding subtree. Otherwise, this simply generates an empty DocumentFragment. DOCUMENT\_NODE Document nodes cannot be imported. DOCUMENT\_TYPE\_NODE DocumentType nodes cannot be imported. ELEMENT\_NODE *Specified* attribute nodes of the source element are imported, and the generated Attr nodes are attached to the generated Element. Default attributes are *not* copied, though if the document being imported into defines default attributes for this element name, those are assigned. If the importNode deep parameter was set to true, the descendants of the source element are recursively imported and the resulting nodes reassembled to form the corresponding subtree. ENTITY\_NODE Entity nodes can be imported, however in the current release of the DOM the DocumentType is readonly. Ability to add these imported nodes to a DocumentType will be considered for addition to a future release of the DOM.On import, the publicId, systemId, and notationName attributes are copied. If a deep import is requested, the descendants of the the source Entity are recursively imported and the resulting nodes reassembled to form the corresponding subtree. ENTITY\_REFERENCE\_NODE Only the EntityReference itself is copied, even if a deep import is requested, since the source and destination documents might have defined the entity differently. If the document being imported into provides a definition for this entity name, its value is assigned. NOTATION\_NODE Notation nodes can be imported, however in the current release of the DOM the DocumentType is readonly. Ability to add these imported nodes to a DocumentType will be considered for addition to a future release of the DOM.On import, the publicId and systemId attributes are copied. Note that the deep parameter has no effect on this type of nodes since they cannot have any children. PROCESSING\_INSTRUCTION\_NODE The imported node copies its target and data values from those of the source node.Note that the deep parameter has no effect on this type of nodes since they cannot have any children. TEXT\_NODE, CDATA\_SECTION\_NODE, COMMENT\_NODE These three types of nodes inheriting from CharacterData copy their data and length attributes from those of the source node.Note that the deep parameter has no effect on these types of nodes since they cannot have any children.

**Parameters:**importedNode - The node to import.deep - If true, recursively import the subtree under the specified node; if false, import only the node itself, as explained above. This has no effect on nodes that cannot have any children, and on Attr, and EntityReference nodes. **Returns:**The imported node that belongs to this Document. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised if the type of node being imported is not supported.

INVALID\_CHARACTER\_ERR: Raised if one of the imported names is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute. This may happen when importing an XML 1.1 [[XML 1.1](http://www.w3.org/TR/2004/REC-xml11-20040204/)] element into an XML 1.0 document, for instance.**Since:** DOM Level 2

### createElementNS

[Element](http://docs.google.com/org/w3c/dom/Element.html) **createElementNS**([String](http://docs.google.com/java/lang/String.html) namespaceURI,  
 [String](http://docs.google.com/java/lang/String.html) qualifiedName)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates an element of the given qualified name and namespace URI.

Per [[XML Namespaces](http://www.w3.org/TR/1999/REC-xml-names-19990114/)] , applications must use the value null as the namespaceURI parameter for methods if they wish to have no namespace.

**Parameters:**namespaceURI - The namespace URI of the element to create.qualifiedName - The qualified name of the element type to instantiate. **Returns:**A new Element object with the following attributes:

| Attribute | Value |
| --- | --- |
| Node.nodeName | qualifiedName |
| Node.namespaceURI | namespaceURI |
| Node.prefix | prefix, extracted from qualifiedName, or null if there is no prefix |
| Node.localName | local name, extracted from qualifiedName |
| Element.tagName | qualifiedName |

**Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified qualifiedName is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

NAMESPACE\_ERR: Raised if the qualifiedName is a malformed qualified name, if the qualifiedName has a prefix and the namespaceURI is null, or if the qualifiedName has a prefix that is "xml" and the namespaceURI is different from "<http://www.w3.org/XML/1998/namespace>" [[XML Namespaces](http://www.w3.org/TR/1999/REC-xml-names-19990114/)] , or if the qualifiedName or its prefix is "xmlns" and the namespaceURI is different from "<http://www.w3.org/2000/xmlns/>", or if the namespaceURI is "<http://www.w3.org/2000/xmlns/>" and neither the qualifiedName nor its prefix is "xmlns".

NOT\_SUPPORTED\_ERR: Always thrown if the current document does not support the "XML" feature, since namespaces were defined by XML.**Since:** DOM Level 2

### createAttributeNS

[Attr](http://docs.google.com/org/w3c/dom/Attr.html) **createAttributeNS**([String](http://docs.google.com/java/lang/String.html) namespaceURI,  
 [String](http://docs.google.com/java/lang/String.html) qualifiedName)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Creates an attribute of the given qualified name and namespace URI.

Per [[XML Namespaces](http://www.w3.org/TR/1999/REC-xml-names-19990114/)] , applications must use the value null as the namespaceURI parameter for methods if they wish to have no namespace.

**Parameters:**namespaceURI - The namespace URI of the attribute to create.qualifiedName - The qualified name of the attribute to instantiate. **Returns:**A new Attr object with the following attributes:

| Attribute | Value |
| --- | --- |
| Node.nodeName | qualifiedName |
| Node.namespaceURI | namespaceURI |
| Node.prefix | prefix, extracted from qualifiedName, or null if there is no prefix |
| Node.localName | local name, extracted from qualifiedName |
| Attr.name | qualifiedName |
| Node.nodeValue | the empty string |

**Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - INVALID\_CHARACTER\_ERR: Raised if the specified qualifiedName is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

NAMESPACE\_ERR: Raised if the qualifiedName is a malformed qualified name, if the qualifiedName has a prefix and the namespaceURI is null, if the qualifiedName has a prefix that is "xml" and the namespaceURI is different from "<http://www.w3.org/XML/1998/namespace>", if the qualifiedName or its prefix is "xmlns" and the namespaceURI is different from "<http://www.w3.org/2000/xmlns/>", or if the namespaceURI is "<http://www.w3.org/2000/xmlns/>" and neither the qualifiedName nor its prefix is "xmlns".

NOT\_SUPPORTED\_ERR: Always thrown if the current document does not support the "XML" feature, since namespaces were defined by XML.**Since:** DOM Level 2

### getElementsByTagNameNS

[NodeList](http://docs.google.com/org/w3c/dom/NodeList.html) **getElementsByTagNameNS**([String](http://docs.google.com/java/lang/String.html) namespaceURI,  
 [String](http://docs.google.com/java/lang/String.html) localName)

Returns a NodeList of all the Elements with a given local name and namespace URI in document order.

**Parameters:**namespaceURI - The namespace URI of the elements to match on. The special value "\*" matches all namespaces.localName - The local name of the elements to match on. The special value "\*" matches all local names. **Returns:**A new NodeList object containing all the matched Elements.**Since:** DOM Level 2

### getElementById

[Element](http://docs.google.com/org/w3c/dom/Element.html) **getElementById**([String](http://docs.google.com/java/lang/String.html) elementId)

Returns the Element that has an ID attribute with the given value. If no such element exists, this returns null . If more than one element has an ID attribute with that value, what is returned is undefined.

The DOM implementation is expected to use the attribute Attr.isId to determine if an attribute is of type ID.

**Note:** Attributes with the name "ID" or "id" are not of type ID unless so defined.

**Parameters:**elementId - The unique id value for an element. **Returns:**The matching element or null if there is none.**Since:** DOM Level 2

### getInputEncoding

[String](http://docs.google.com/java/lang/String.html) **getInputEncoding**()

An attribute specifying the encoding used for this document at the time of the parsing. This is null when it is not known, such as when the Document was created in memory.

**Since:** DOM Level 3

### getXmlEncoding

[String](http://docs.google.com/java/lang/String.html) **getXmlEncoding**()

An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the encoding of this document. This is null when unspecified or when it is not known, such as when the Document was created in memory.

**Since:** DOM Level 3

### getXmlStandalone

boolean **getXmlStandalone**()

An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), whether this document is standalone. This is false when unspecified.

**Note:** No verification is done on the value when setting this attribute. Applications should use Document.normalizeDocument() with the "validate" parameter to verify if the value matches the [validity constraint for standalone document declaration](http://www.w3.org/TR/2004/REC-xml-20040204#sec-rmd) as defined in [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)].

**Since:** DOM Level 3

### setXmlStandalone

void **setXmlStandalone**(boolean xmlStandalone)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), whether this document is standalone. This is false when unspecified.

**Note:** No verification is done on the value when setting this attribute. Applications should use Document.normalizeDocument() with the "validate" parameter to verify if the value matches the [validity constraint for standalone document declaration](http://www.w3.org/TR/2004/REC-xml-20040204#sec-rmd) as defined in [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)].

**Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised if this document does not support the "XML" feature.**Since:** DOM Level 3

### getXmlVersion

[String](http://docs.google.com/java/lang/String.html) **getXmlVersion**()

An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the version number of this document. If there is no declaration and if this document supports the "XML" feature, the value is "1.0". If this document does not support the "XML" feature, the value is always null. Changing this attribute will affect methods that check for invalid characters in XML names. Application should invoke Document.normalizeDocument() in order to check for invalid characters in the Nodes that are already part of this Document.

DOM applications may use the DOMImplementation.hasFeature(feature, version) method with parameter values "XMLVersion" and "1.0" (respectively) to determine if an implementation supports [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)]. DOM applications may use the same method with parameter values "XMLVersion" and "1.1" (respectively) to determine if an implementation supports [[XML 1.1](http://www.w3.org/TR/2004/REC-xml11-20040204/)]. In both cases, in order to support XML, an implementation must also support the "XML" feature defined in this specification. Document objects supporting a version of the "XMLVersion" feature must not raise a NOT\_SUPPORTED\_ERR exception for the same version number when using Document.xmlVersion.

**Since:** DOM Level 3

### setXmlVersion

void **setXmlVersion**([String](http://docs.google.com/java/lang/String.html) xmlVersion)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

An attribute specifying, as part of the [XML declaration](http://www.w3.org/TR/2004/REC-xml-20040204#NT-XMLDecl), the version number of this document. If there is no declaration and if this document supports the "XML" feature, the value is "1.0". If this document does not support the "XML" feature, the value is always null. Changing this attribute will affect methods that check for invalid characters in XML names. Application should invoke Document.normalizeDocument() in order to check for invalid characters in the Nodes that are already part of this Document.

DOM applications may use the DOMImplementation.hasFeature(feature, version) method with parameter values "XMLVersion" and "1.0" (respectively) to determine if an implementation supports [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)]. DOM applications may use the same method with parameter values "XMLVersion" and "1.1" (respectively) to determine if an implementation supports [[XML 1.1](http://www.w3.org/TR/2004/REC-xml11-20040204/)]. In both cases, in order to support XML, an implementation must also support the "XML" feature defined in this specification. Document objects supporting a version of the "XMLVersion" feature must not raise a NOT\_SUPPORTED\_ERR exception for the same version number when using Document.xmlVersion.

**Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised if the version is set to a value that is not supported by this Document or if this document does not support the "XML" feature.**Since:** DOM Level 3

### getStrictErrorChecking

boolean **getStrictErrorChecking**()

An attribute specifying whether error checking is enforced or not. When set to false, the implementation is free to not test every possible error case normally defined on DOM operations, and not raise any DOMException on DOM operations or report errors while using Document.normalizeDocument(). In case of error, the behavior is undefined. This attribute is true by default.

**Since:** DOM Level 3

### setStrictErrorChecking

void **setStrictErrorChecking**(boolean strictErrorChecking)

An attribute specifying whether error checking is enforced or not. When set to false, the implementation is free to not test every possible error case normally defined on DOM operations, and not raise any DOMException on DOM operations or report errors while using Document.normalizeDocument(). In case of error, the behavior is undefined. This attribute is true by default.

**Since:** DOM Level 3

### getDocumentURI

[String](http://docs.google.com/java/lang/String.html) **getDocumentURI**()

The location of the document or null if undefined or if the Document was created using DOMImplementation.createDocument. No lexical checking is performed when setting this attribute; this could result in a null value returned when using Node.baseURI .

Beware that when the Document supports the feature "HTML" [[DOM Level 2 HTML](http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109)] , the href attribute of the HTML BASE element takes precedence over this attribute when computing Node.baseURI.

**Since:** DOM Level 3

### setDocumentURI

void **setDocumentURI**([String](http://docs.google.com/java/lang/String.html) documentURI)

The location of the document or null if undefined or if the Document was created using DOMImplementation.createDocument. No lexical checking is performed when setting this attribute; this could result in a null value returned when using Node.baseURI .

Beware that when the Document supports the feature "HTML" [[DOM Level 2 HTML](http://www.w3.org/TR/2003/REC-DOM-Level-2-HTML-20030109)] , the href attribute of the HTML BASE element takes precedence over this attribute when computing Node.baseURI.

**Since:** DOM Level 3

### adoptNode

[Node](http://docs.google.com/org/w3c/dom/Node.html) **adoptNode**([Node](http://docs.google.com/org/w3c/dom/Node.html) source)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Attempts to adopt a node from another document to this document. If supported, it changes the ownerDocument of the source node, its children, as well as the attached attribute nodes if there are any. If the source node has a parent it is first removed from the child list of its parent. This effectively allows moving a subtree from one document to another (unlike importNode() which create a copy of the source node instead of moving it). When it fails, applications should use Document.importNode() instead. Note that if the adopted node is already part of this document (i.e. the source and target document are the same), this method still has the effect of removing the source node from the child list of its parent, if any. The following list describes the specifics for each type of node. ATTRIBUTE\_NODE The ownerElement attribute is set to null and the specified flag is set to true on the adopted Attr. The descendants of the source Attr are recursively adopted. DOCUMENT\_FRAGMENT\_NODE The descendants of the source node are recursively adopted. DOCUMENT\_NODE Document nodes cannot be adopted. DOCUMENT\_TYPE\_NODE DocumentType nodes cannot be adopted. ELEMENT\_NODE *Specified* attribute nodes of the source element are adopted. Default attributes are discarded, though if the document being adopted into defines default attributes for this element name, those are assigned. The descendants of the source element are recursively adopted. ENTITY\_NODE Entity nodes cannot be adopted. ENTITY\_REFERENCE\_NODE Only the EntityReference node itself is adopted, the descendants are discarded, since the source and destination documents might have defined the entity differently. If the document being imported into provides a definition for this entity name, its value is assigned. NOTATION\_NODE Notation nodes cannot be adopted. PROCESSING\_INSTRUCTION\_NODE, TEXT\_NODE, CDATA\_SECTION\_NODE, COMMENT\_NODE These nodes can all be adopted. No specifics.

**Note:** Since it does not create new nodes unlike the Document.importNode() method, this method does not raise an INVALID\_CHARACTER\_ERR exception, and applications should use the Document.normalizeDocument() method to check if an imported name is not an XML name according to the XML version in use.

**Parameters:**source - The node to move into this document. **Returns:**The adopted node, or null if this operation fails, such as when the source node comes from a different implementation. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised if the source node is of type DOCUMENT, DOCUMENT\_TYPE.

NO\_MODIFICATION\_ALLOWED\_ERR: Raised when the source node is readonly.**Since:** DOM Level 3

### getDomConfig

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

The configuration used when Document.normalizeDocument() is invoked.

**Since:** DOM Level 3

### normalizeDocument

void **normalizeDocument**()

This method acts as if the document was going through a save and load cycle, putting the document in a "normal" form. As a consequence, this method updates the replacement tree of EntityReference nodes and normalizes Text nodes, as defined in the method Node.normalize().

Otherwise, the actual result depends on the features being set on the Document.domConfig object and governing what operations actually take place. Noticeably this method could also make the document namespace well-formed according to the algorithm described in , check the character normalization, remove the CDATASection nodes, etc. See DOMConfiguration for details.

// Keep in the document   
 the information defined // in the XML Information Set (Java example)   
 DOMConfiguration docConfig = myDocument.getDomConfig();   
 docConfig.setParameter("infoset", Boolean.TRUE);   
 myDocument.normalizeDocument();

Mutation events, when supported, are generated to reflect the changes occurring on the document.

If errors occur during the invocation of this method, such as an attempt to update a read-only node or a Node.nodeName contains an invalid character according to the XML version in use, errors or warnings (DOMError.SEVERITY\_ERROR or DOMError.SEVERITY\_WARNING) will be reported using the DOMErrorHandler object associated with the "error-handler " parameter. Note this method might also report fatal errors ( DOMError.SEVERITY\_FATAL\_ERROR) if an implementation cannot recover from an error.

**Since:** DOM Level 3

### renameNode

[Node](http://docs.google.com/org/w3c/dom/Node.html) **renameNode**([Node](http://docs.google.com/org/w3c/dom/Node.html) n,  
 [String](http://docs.google.com/java/lang/String.html) namespaceURI,  
 [String](http://docs.google.com/java/lang/String.html) qualifiedName)  
 throws [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html)

Rename an existing node of type ELEMENT\_NODE or ATTRIBUTE\_NODE.

When possible this simply changes the name of the given node, otherwise this creates a new node with the specified name and replaces the existing node with the new node as described below.

If simply changing the name of the given node is not possible, the following operations are performed: a new node is created, any registered event listener is registered on the new node, any user data attached to the old node is removed from that node, the old node is removed from its parent if it has one, the children are moved to the new node, if the renamed node is an Element its attributes are moved to the new node, the new node is inserted at the position the old node used to have in its parent's child nodes list if it has one, the user data that was attached to the old node is attached to the new node.

When the node being renamed is an Element only the specified attributes are moved, default attributes originated from the DTD are updated according to the new element name. In addition, the implementation may update default attributes from other schemas. Applications should use Document.normalizeDocument() to guarantee these attributes are up-to-date.

When the node being renamed is an Attr that is attached to an Element, the node is first removed from the Element attributes map. Then, once renamed, either by modifying the existing node or creating a new one as described above, it is put back.

In addition,

* a user data event NODE\_RENAMED is fired,
* when the implementation supports the feature "MutationNameEvents", each mutation operation involved in this method fires the appropriate event, and in the end the event { http://www.w3.org/2001/xml-events, DOMElementNameChanged} or { http://www.w3.org/2001/xml-events, DOMAttributeNameChanged} is fired.

**Parameters:**n - The node to rename.namespaceURI - The new namespace URI.qualifiedName - The new qualified name. **Returns:**The renamed node. This is either the specified node or the new node that was created to replace the specified node. **Throws:** [DOMException](http://docs.google.com/org/w3c/dom/DOMException.html) - NOT\_SUPPORTED\_ERR: Raised when the type of the specified node is neither ELEMENT\_NODE nor ATTRIBUTE\_NODE, or if the implementation does not support the renaming of the document element.

INVALID\_CHARACTER\_ERR: Raised if the new qualified name is not an XML name according to the XML version in use specified in the Document.xmlVersion attribute.

WRONG\_DOCUMENT\_ERR: Raised when the specified node was created from a different document than this document.

NAMESPACE\_ERR: Raised if the qualifiedName is a malformed qualified name, if the qualifiedName has a prefix and the namespaceURI is null, or if the qualifiedName has a prefix that is "xml" and the namespaceURI is different from "<http://www.w3.org/XML/1998/namespace>" [[XML Namespaces](http://www.w3.org/TR/1999/REC-xml-names-19990114/)] . Also raised, when the node being renamed is an attribute, if the qualifiedName, or its prefix, is "xmlns" and the namespaceURI is different from "<http://www.w3.org/2000/xmlns/>".**Since:** DOM Level 3

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Document.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/Comment.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/DocumentFragment.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/Document.html)    [**NO FRAMES**](http://docs.google.com/Document.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#1t3h5sf) |

[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).