

---

# NSXMLElement Class Reference

Data Management: Data Types & Collections



2007-02-27



Apple Inc.  
© 2007 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014  
408-996-1010

Apple, the Apple logo, Cocoa, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY**

**DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

## NSXMLElement Class Reference 5

---

Overview	5
Subclassing Notes	5
Tasks	6
Initializing NSXMLElement Objects	6
Obtaining Child Elements	7
Manipulating Child Elements	7
Handling Attributes	7
Handling Namespaces	8
Instance Methods	8
addAttribute:	8
addChild:	9
addNamespace:	9
attributeForLocalName:URI:	10
attributeForName:	10
attributes	11
elementsForLocalName:URI:	11
elementsForName:	12
initWithName:	12
initWithName:stringValue:	13
initWithName:URI:	13
initWithXMLString:error:	14
insertChild:atIndex:	14
insertChildren:atIndex:	15
namespaceForPrefix:	16
namespaces	16
normalizeAdjacentTextNodesPreservingCDATA:	17
removeAttributeForName:	17
removeChildAtIndex:	18
removeNamespaceForPrefix:	18
replaceChildAtIndex:withNode:	19
resolveNamespaceForName:	19
resolvePrefixForNamespaceURI:	20
setAttributes:	20
setAttributesAsDictionary:	21
setChildren:	21
setNamespaces:	22

**Document Revision History 23**

---

**Index 25**

---

# NSXMLElement Class Reference

---

<b>Inherits from</b>	NSXMLNode : NSObject
<b>Conforms to</b>	NSCopying (NSXMLNode) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Foundation.framework
<b>Availability</b>	Available in Mac OS X v10.4 and later.
<b>Companion guide</b>	Tree-Based XML Programming Guide for Cocoa
<b>Declared in</b>	NSXMLElement.h
<b>Related sample code</b>	AlbumToSlideshow Core Data HTML Store MovieAssembler TimelineToTC XMLBrowser

## Overview

Instances of the `NSXMLElement` class represent element nodes in an XML tree structure. An `NSXMLElement` object may have child nodes, specifically comment nodes, processing-instruction nodes, text nodes, and other `NSXMLElement` nodes. It may also have attribute nodes and namespace nodes associated with it (however, namespace and attribute nodes are not considered children). Any attempt to add a `NSXMLDocument` node, `NSXMLDTD` node, namespace node, or attribute node as a child raises an exception. If you add a child node to an `NSXMLElement` object and that child already has a parent, `NSXMLElement` raises an exception; the child must be detached or copied first.

## Subclassing Notes

---

You can subclass `NSXMLElement` if you want element nodes with more specialized attributes or behavior, for example, paragraph and font attributes that specify how the string value of the element should appear.

## Methods to Override

---

To subclass `NSXMLElement` you need to override the primary initializer, `initWithName:URI:` (page 13), and the methods listed below. In most cases, you need only invoke the superclass implementation, adding any subclass-specific code before or after the invocation, as necessary.

<a href="#">addAttribute:</a> (page 8)	<a href="#">removeNamespaceForPrefix:</a> (page 18)
<a href="#">removeAttributeForName:</a> (page 17)	<a href="#">setNamespaces:</a> (page 22)
<a href="#">setAttributes:</a> (page 20)	<a href="#">namespaces</a> (page 16)
<a href="#">attributeForLocalName:URI:</a> (page 10)	<a href="#">insertChildAtIndex:</a> (page 14)
<a href="#">attributes</a> (page 11)	<a href="#">removeChildAtIndex:</a> (page 18)
<a href="#">addNamespace:</a> (page 9)	<a href="#">setChildren:</a> (page 21)

By default `NSXMLElement` implements the `NSObject isEqual:` method to perform a deep comparison: two `NSXMLDocument` objects are not considered equal unless they have the same name, same child nodes, same attributes, and so on. If you want a different standard of comparison, override `isEqual:`.

## Special Considerations

---

Because of the architecture and data model of NSXML, when it parses and processes a source of XML it cannot know about your subclass unless you override the class method `replacementClassForClass:` to return your custom class in place of an NSXML class. If your custom class has no direct NSXML counterpart—for example, it is a subclass of `NSXMLNode` that represents CDATA sections—then you can walk the tree after it has been created and insert the new node where appropriate.

Note that you can safely set the root element of the XML document (using the `NSXMLDocument setRootElement:` method) to be an instance of your subclass since this method only checks to see if the added node is of an element kind (`NSXMLElementKind`). These precautions do not apply, of course, if you are creating an XML tree programmatically.

## Tasks

### Initializing NSXMLElement Objects

- [initWithName:](#) (page 12)  
Returns an `NSXMLElement` object initialized with the specified name.
- [initWithName:stringValue:](#) (page 13)  
Returns an `NSXMLElement` object initialized with a specified name and a single text-node child containing a specified value.
- [initWithXMLString:error:](#) (page 14)  
Returns an `NSXMLElement` object created from a specified string containing XML markup.
- [initWithName:URI:](#) (page 13)  
Returns an `NSXMLElement` object initialized with the specified name and URI.

## Obtaining Child Elements

- [elementsForName:](#) (page 12)  
Returns the child element nodes (as `NSXMLElement` objects) of the receiver that have a specified name.
- [elementsForLocalName:URI:](#) (page 11)  
Returns the child element nodes (as `NSXMLElement` objects) of the receiver that are matched with the specified local name and URI.

## Manipulating Child Elements

- [addChild:](#) (page 9)  
Adds a child node at the end of the receiver's current list of children.
- [insertChild:atIndex:](#) (page 14)  
Inserts a new child node at a specified location in the receiver's list of child nodes.
- [insertChildren:atIndex:](#) (page 15)  
Inserts an array of child nodes at a specified location in the receiver's list of children.
- [removeChildAtIndex:](#) (page 18)  
Removes the child node of the receiver identified by a given index.
- [replaceChildAtIndex:withNode:](#) (page 19)  
Replaces a child node at a specified location with another child node.
- [setChildren:](#) (page 21)  
Sets all child nodes of the receiver at once, replacing any existing children.
- [normalizeAdjacentTextNodesPreservingCDATA:](#) (page 17)  
Coalesces adjacent text nodes of the receiver that you have explicitly added, optionally including CDATA sections.

## Handling Attributes

- [addAttribute:](#) (page 8)  
Adds an attribute node to the receiver.
- [attributeForName:](#) (page 10)  
Returns the attribute node of the receiver with the specified name.
- [attributeForLocalName:URI:](#) (page 10)  
Returns the attribute node of the receiver that is identified by a local name and URI.
- [attributes](#) (page 11)  
Returns the receiver's attributes
- [removeAttributeForName:](#) (page 17)  
Removes an attribute node that is identified by its name.
- [setAttributes:](#) (page 20)  
Sets all attributes of the receiver at once, replacing any existing attribute nodes.
- [setAttributesAsDictionary:](#) (page 21)  
Sets the attributes of the receiver based on the key-value pairs specified in the passed-in dictionary.

## Handling Namespaces

- [addNamespace:](#) (page 9)  
Adds a namespace node to the receiver.
- [namespaces](#) (page 16)  
Returns the namespace nodes of the receiver.
- [namespaceForPrefix:](#) (page 16)  
Returns the namespace node with a specified prefix.
- [removeNamespaceForPrefix:](#) (page 18)  
Removes a namespace node that is identified by a given prefix.
- [resolveNamespaceForName:](#) (page 19)  
Returns the namespace node with the prefix matching the given qualified name.
- [resolvePrefixForNamespaceURI:](#) (page 20)  
Returns the prefix associated with the specified URI.
- [setNamespaces:](#) (page 22)  
Sets all of the namespace nodes of the receiver at once, replacing any existing namespace nodes.

## Instance Methods

### addAttribute:

Adds an attribute node to the receiver.

```
- (void)addAttribute:(NSXMLNode *)anAttribute
```

#### Parameters

*anAttribute*

An XML node object representing an attribute. If the receiver already has an attribute with the same name, *anAttribute* is not added.

#### Discussion

The order of multiple attributes is preserved if the `NSXMLPreserveAttributeOrder` option is specified when the element is created.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [attributeForName:](#) (page 10)
- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

#### Related Sample Code

AlbumToSlideshow

Core Data HTML Store

**Declared In**

NSXMLElement.h

**addChild:**

Adds a child node at the end of the receiver's current list of children.

```
- (void)addChild:(NSXMLNode *)child
```

**Parameters***child*

An XML node object to add to the receiver's children.

**Discussion**

The new node has an index value that is one greater than the last of the current children.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [insertChildAtIndex:](#) (page 14)
- [removeChildAtIndex:](#) (page 18)
- [replaceChildAtIndex:withNode:](#) (page 19)
- [setChildren:](#) (page 21)

**Related Sample Code**

AlbumToSlideshow

Core Data HTML Store

MovieAssembler

SimpleScriptingPlugin

**Declared In**

NSXMLElement.h

**addNamespace:**

Adds a namespace node to the receiver.

```
- (void)addNamespace:(NSXMLNode *)aNamespace
```

**Parameters***aNamespace*

An XML node object of kind `NSXMLNamespaceKind`. If the receiver already has a namespace with the same name, *aNamespace* is not added.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [namespaces](#) (page 16)
- [namespaceForPrefix:](#) (page 16)
- [removeNamespaceForPrefix:](#) (page 18)

- [resolveNamespaceForName:](#) (page 19)
- [resolvePrefixForNamespaceURI:](#) (page 20)
- [setNamespaces:](#) (page 22)

**Declared In**

NSXMLElement.h

**attributeForLocalName:URI:**

Returns the attribute node of the receiver that is identified by a local name and URI.

```
-(NSXMLNode *)attributeForLocalName:(NSString *)localName URI:(NSString *)URI
```

**Parameters***localName*

A string specifying the local name of an attribute.

*URI*

A string identifying the URI associated with an attribute.

**Return Value**

An XML node object representing a matching attribute or `nil` if no such node was found.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [attributeForName:](#) (page 10)
- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

**Declared In**

NSXMLElement.h

**attributeForName:**

Returns the attribute node of the receiver with the specified name.

```
-(NSXMLNode *)attributeForName:(NSString *)name
```

**Parameters***name*

A string specifying the name of an attribute.

**Return Value**

An XML node object representing a matching attribute or `nil` if no such node was found.

**Discussion**

If *name* is a qualified name, then this method invokes [attributeForLocalName:URI:](#) (page 10) with the URI parameter set to the URI associated with the prefix. Otherwise comparison is based on string equality of the qualified or non-qualified name.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

**Related Sample Code**

Core Data HTML Store

**Declared In**

NSXMLElement.h

**attributes**

Returns the receiver's attributes

- (NSArray \*)attributes

**Return Value**

An array of `NSXMLNode` objects of kind `NSXMLAttributeKind` or `nil` if the receiver has no attribute nodes.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [attributeForLocalName:URI:](#) (page 10)
- [attributeForName:](#) (page 10)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

**Declared In**

NSXMLElement.h

**elementsForLocalName:URI:**

Returns the child element nodes (as `NSXMLElement` objects) of the receiver that are matched with the specified local name and URI.

- (NSArray \*)elementsForLocalName:(NSString \*)*localName* URI:(NSString \*)*URI*

**Parameters**

*localName*

A string specifying a local name of an element.

*URI*

A string specifying a URI associated with an element.

**Return Value**

An array of `NSXMLElement` objects or `nil` if no matching children could be found.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [elementsForName:](#) (page 12)

**Declared In**

NSXMLElement.h

**elementsForName:**

Returns the child element nodes (as `NSXMLElement` objects) of the receiver that have a specified name.

```
- (NSArray *)elementsForName:(NSString *)name
```

**Parameters**

*name*

A string specifying the name of the child element nodes to find and return. If *name* is a qualified name, then this method invokes [elementsForLocalName:URI:](#) (page 11) with the URI parameter set to the URI associated with the prefix. Otherwise comparison is based on string equality of the qualified or non-qualified name.

**Return Value**

An array of of `NSXMLElement` objects or an empty array if no matching children can be found.

**Availability**

Available in Mac OS X v10.4 and later.

**Related Sample Code**

Core Data HTML Store

NewsReader

**Declared In**

NSXMLElement.h

**initWithName:**

Returns an `NSXMLElement` object initialized with the specified name.

```
- (id)initWithName:(NSString *)name
```

**Parameters**

*name*

A string specifying the name of the element.

**Return Value**

The initialized `NSXMLElement` object or `nil` if initialization did not succeed.

**Discussion**

The XML string representation of this object is `<name></name>`. This method invokes [initWithName:URI:](#) (page 13) with the URI parameter set to `nil`.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [initWithName:stringValue:](#) (page 13)
- [initWithXMLString:error:](#) (page 14)

**Declared In**

NSXMLElement.h

**initWithName:stringValue:**

Returns an `NSXMLElement` object initialized with a specified name and a single text-node child containing a specified value.

```
- (id)initWithName:(NSString *)name stringValue:(NSString *)string
```

**Parameters***name*

A string specifying the name of the element.

*string*

The string value of the receiver's text node.

**Return Value**

The initialized `NSXMLElement` object or `nil` if initialization did not succeed.

**Discussion**

The string representation of this object is `<name>string</name>`.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [initWithName:URI:](#) (page 13)
- [initWithName:](#) (page 12)
- [initWithXMLString:error:](#) (page 14)

**Declared In**

NSXMLElement.h

**initWithName:URI:**

Returns an `NSXMLElement` object initialized with the specified name and URI.

```
- (id)initWithName:(NSString *)name URI:(NSString *)URI
```

**Parameters***name*

A string that specifies the qualified name of the element.

*URI*

A string that specifies the namespace URI associated with the element.

**Return Value**

The initialized `NSXMLElement` object or `nil` if initialization did not succeed.

**Discussion**

You can look up the namespace prefix for this element node based on its URI using [resolvePrefixForNamespaceURI:](#) (page 20). This method is the primary initializer for the `NSXMLElement` class.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [initWithName:](#) (page 12)
- [initWithName:stringValue:](#) (page 13)
- [initWithXMLString:error:](#) (page 14)

**Declared In**

`NSXMLElement.h`

**initWithXMLString:error:**

Returns an `NSXMLElement` object created from a specified string containing XML markup.

```
- (id)initWithXMLString:(NSString *)string error:(NSError **)error
```

**Parameters**

*string*

A string containing XML markup for an element.

*error*

On return, an `NSError` object that describes any errors or warnings resulting from the parsing of the markup.

**Return Value**

The initialized `NSXMLElement` object or `nil` if initialization did not succeed.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [initWithName:URI:](#) (page 13)
- [initWithName:](#) (page 12)
- [initWithName:stringValue:](#) (page 13)

**Declared In**

`NSXMLElement.h`

**insertChild:atIndex:**

Inserts a new child node at a specified location in the receiver's list of child nodes.

```
- (void)insertChild:(NSXMLNode *)child atIndex:(NSUInteger)index
```

**Parameters**

*child*

An XML node object to be inserted as a child of the receiver.

*index*

An integer identifying a position in the receiver's list of children. An exception is raised if *index* is out of bounds.

#### Discussion

Insertion of the node increments the indexes of sibling nodes after it.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [addChild:](#) (page 9)
- [insertChildren:atIndex:](#) (page 15)
- [removeChildAtIndex:](#) (page 18)
- [replaceChildAtIndex:withNode:](#) (page 19)
- [setChildren:](#) (page 21)

#### Related Sample Code

MovieAssembler

#### Declared In

NSXMLElement.h

## insertChildren:atIndex:

Inserts an array of child nodes at a specified location in the receiver's list of children.

```
- (void)insertChildren:(NSArray *)children atIndex:(NSUInteger)index
```

#### Parameters

*children*

An array of XML node objects to add as children of the receiver.

*index*

An integer identifying a position in the receiver's list of children. An exception is raised if *index* is out of bounds.

#### Discussion

Insertion of the node increases the indexes of sibling nodes after it by the count of *children*.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [addChild:](#) (page 9)
- [insertChild:atIndex:](#) (page 14)
- [removeChildAtIndex:](#) (page 18)
- [replaceChildAtIndex:withNode:](#) (page 19)
- [setChildren:](#) (page 21)

#### Declared In

NSXMLElement.h

## namespaceForPrefix:

Returns the namespace node with a specified prefix.

- (NSXMLElement \*)namespaceForPrefix:(NSString \*)name

### Parameters

*name*

A string specifying a namespace prefix.

### Return Value

An NSXMLElement object of kind NSXMLNamespaceKind or nil if there is no namespace node with that prefix.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addNamespace:](#) (page 9)
- [namespaces](#) (page 16)
- [removeNamespaceForPrefix:](#) (page 18)
- [resolveNamespaceForName:](#) (page 19)
- [resolvePrefixForNamespaceURI:](#) (page 20)
- [setNamespaces:](#) (page 22)

### Declared In

NSXMLElement.h

## namespaces

Returns the namespace nodes of the receiver.

- (NSArray \*)namespaces

### Return Value

An array of NSXMLElement objects of kind NSXMLNamespaceKind. Returns nil if the receiver has no namespace nodes.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addNamespace:](#) (page 9)
- [namespaceForPrefix:](#) (page 16)
- [removeNamespaceForPrefix:](#) (page 18)
- [resolveNamespaceForName:](#) (page 19)
- [resolvePrefixForNamespaceURI:](#) (page 20)
- [setNamespaces:](#) (page 22)

### Declared In

NSXMLElement.h

## normalizeAdjacentTextNodesPreservingCDATA:

Coalesces adjacent text nodes of the receiver that you have explicitly added, optionally including CDATA sections.

```
- (void)normalizeAdjacentTextNodesPreservingCDATA:(BOOL)preserve
```

### Parameters

*preserve*

YES if CDATA sections are left alone as text nodes, NO otherwise.

### Discussion

A text node with a value of an empty string is removed. When you process an input source of XML, adjacent text nodes are automatically normalized. You should invoke this method (with *preserve* as NO) before using the NSXMLNode methods `objectsForXPath:error:` or `nodesForXPath:error:`.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setChildren:](#) (page 21)

### Declared In

NSXMLElement.h

## removeAttributeForName:

Removes an attribute node that is identified by its name.

```
- (void)removeAttributeForName:(NSString *)attrName
```

### Parameters

*attrName*

A string specifying the name of an attribute.

### Discussion

The removed XML node object is released.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addAttribute:](#) (page 8)
- [attributeForName:](#) (page 10)
- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

### Declared In

NSXMLElement.h

## removeChildAtIndex:

Removes the child node of the receiver identified by a given index.

- (void)removeChildAtIndex:(NSUInteger)*nodeIndex*

### Parameters

*nodeIndex*

An integer identifying the node in the receiver's list of children to remove. An exception is raised if *index* is out of bounds.

### Discussion

The XML node object is released upon removal. The indices of subsequent children are decremented by one.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addChild:](#) (page 9)
- [insertChildAtIndex:](#) (page 14)
- [replaceChildAtIndex:withNode:](#) (page 19)
- [setChildren:](#) (page 21)

### Related Sample Code

Core Data HTML Store

### Declared In

NSXMLElement.h

## removeNamespaceForPrefix:

Removes a namespace node that is identified by a given prefix.

- (void)removeNamespaceForPrefix:(NSString \*)*name*

### Parameters

*name*

A string that is the prefix for a namespace.

### Discussion

The removed XML node object is removed.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addNamespace:](#) (page 9)
- [namespaces](#) (page 16)
- [namespaceForPrefix:](#) (page 16)
- [setNamespaces:](#) (page 22)

### Declared In

NSXMLElement.h

## replaceChildAtIndex:withNode:

Replaces a child node at a specified location with another child node.

```
- (void)replaceChildAtIndex:(NSUInteger) index withNode:(NSXMLNode *) node
```

### Parameters

*index*

An integer identifying a position in the receiver's list of children. An exception is raised if *index* is out of bounds.

*node*

An XML node object that will replace the current child.

### Discussion

The replaced XML node object is released upon removal.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addChild:](#) (page 9)
- [insertChildAtIndex:](#) (page 14)
- [insertChildrenAtIndex:](#) (page 15)
- [removeChildAtIndex:](#) (page 18)
- [setChildren:](#) (page 21)

### Declared In

NSXMLElement.h

## resolveNamespaceForName:

Returns the namespace node with the prefix matching the given qualified name.

```
- (NSXMLNode *)resolveNamespaceForName:(NSString *) name
```

### Parameters

*name*

A string that is the qualified name for a namespace (a qualified name is prefix plus local name).

### Return Value

An NSXMLNode object of kind NSXMLNamespaceKind or nil if there is no matching namespace node.

### Discussion

The method looks in the entire namespace chain for the prefix.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [addNamespace:](#) (page 9)
- [namespaces](#) (page 16)
- [namespaceForPrefix:](#) (page 16)
- [resolvePrefixForNamespaceURI:](#) (page 20)

- [setNamespaces:](#) (page 22)

#### Declared In

NSXMLElement.h

### resolvePrefixForNamespaceURI:

Returns the prefix associated with the specified URI.

```
- (NSString *)resolvePrefixForNamespaceURI:(NSString *)namespaceURI
```

#### Parameters

*namespaceURI*

A string identifying the URI associated with the namespace.

#### Return Value

A string that is the matching prefix or `nil` if it finds no matching prefix.

#### Discussion

The method looks in the entire namespace chain for the URI.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [addNamespace:](#) (page 9)
- [namespaces](#) (page 16)
- [namespaceForPrefix:](#) (page 16)
- [resolveNamespaceForName:](#) (page 19)
- [setNamespaces:](#) (page 22)

#### Declared In

NSXMLElement.h

### setAttributes:

Sets all attributes of the receiver at once, replacing any existing attribute nodes.

```
- (void)setAttributes:(NSArray *)attributes
```

#### Parameters

*attributes*

An array of `NSXMLNode` objects of kind `NSXMLAttributeKind`. If there are attribute nodes with the same name, the first attribute with that name is used. Send this message with *attributes* as `nil` to remove all attributes.

#### Discussion

To set attributes in an element node using an `NSDictionary` object as the input parameter, see [setAttributesAsDictionary:](#) (page 21).

#### Availability

Available in Mac OS X v10.4 and later.

**See Also**

- [addAttribute:](#) (page 8)
- [attributeForName:](#) (page 10)
- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)

**Declared In**

NSXMLElement.h

**setAttributesAsDictionary:**

Sets the attributes of the receiver based on the key-value pairs specified in the passed-in dictionary.

```
- (void)setAttributesAsDictionary:(NSDictionary *)attributes
```

**Parameters***attributes*

A dictionary of key-value pairs where the attribute name is the key and the object value of the attribute is the dictionary value.

**Discussion**

The method uses these names and object values to create `NSXMLElement` objects of kind `NSXMLElementKind`. Existing attributes are removed.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [addAttribute:](#) (page 8)
- [attributes](#) (page 11)
- [removeAttributeForName:](#) (page 17)
- [setAttributes:](#) (page 20)

**Declared In**

NSXMLElement.h

**setChildren:**

Sets all child nodes of the receiver at once, replacing any existing children.

```
- (void)setChildren:(NSArray *)children
```

**Parameters***children*

An array of `NSXMLElement` objects or `NSXMLElement` objects of kinds `NSXMLElementKind`, `NSXMLProcessingInstructionKind`, `NSXMLTextKind`, or `NSXMLCommentKind`.

**Discussion**

Send this message with *children* as `nil` to remove all child nodes.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [insertChildAtIndex:](#) (page 14)
- [insertChildrenAtIndex:](#) (page 15)
- [removeChildAtIndex:](#) (page 18)
- [replaceChildAtIndex:withNode:](#) (page 19)
- [setChildren:](#) (page 21)

**Related Sample Code**

Core Data HTML Store

**Declared In**

NSXMLElement.h

**setNamespaces:**

Sets all of the namespace nodes of the receiver at once, replacing any existing namespace nodes.

```
- (void)setNamespaces:(NSArray *)namespaces
```

**Parameters***namespaces*

An array of NSXMLNode objects of kind NSXMLNamespaceKind. If there are namespace nodes with the same prefix, the first attribute with that prefix is used. Send this message with *namespaces* as *nil* to remove all namespace nodes.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [addNamespace:](#) (page 9)
- [namespaces](#) (page 16)
- [namespaceForPrefix:](#) (page 16)
- [removeNamespaceForPrefix:](#) (page 18)
- [resolveNamespaceForName:](#) (page 19)
- [resolvePrefixForNamespaceURI:](#) (page 20)

**Declared In**

NSXMLElement.h

# Document Revision History

---

This table describes the changes to *NSXMLElement Class Reference*.

Date	Notes
2007-02-27	Clarified description of setChildren:.
2006-11-07	Corrected typo.
2006-05-23	First publication of this content as a separate document. Revised description of elementsForName:.

## REVISION HISTORY

### Document Revision History

# Index

---

## A

---

addAttribute: [instance method 8](#)  
addChild: [instance method 9](#)  
addNamespace: [instance method 9](#)  
attributeForLocalName:URI: [instance method 10](#)  
attributeForName: [instance method 10](#)  
attributes [instance method 11](#)

## E

---

elementsForLocalName:URI: [instance method 11](#)  
elementsForName: [instance method 12](#)

## I

---

initWithName: [instance method 12](#)  
initWithName:stringValue: [instance method 13](#)  
initWithName:URI: [instance method 13](#)  
initWithXMLString:error: [instance method 14](#)  
insertChild:atIndex: [instance method 14](#)  
insertChildren:atIndex: [instance method 15](#)

## N

---

namespaceForPrefix: [instance method 16](#)  
namespaces [instance method 16](#)  
normalizeAdjacentTextNodesPreservingCDATA:  
[instance method 17](#)

## R

---

removeAttributeForName: [instance method 17](#)  
removeChildAtIndex: [instance method 18](#)  
removeNamespaceForPrefix: [instance method 18](#)

replaceChildAtIndex:withNode: [instance method 19](#)  
resolveNamespaceForName: [instance method 19](#)  
resolvePrefixForNamespaceURI: [instance method 20](#)

## S

---

setAttributesAsDictionary: [instance method 21](#)  
setAttributes: [instance method 20](#)  
setChildren: [instance method 21](#)  
setNamespaces: [instance method 22](#)