Beginning JavaScript®, Fifth Edition
By Jeremy McPeak and Paul Wilton
Copyright © 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana

C

W3C DOM Reference

Because JavaScript is primarily used to program the browser and add behavior to web pages, it's only natural to include a reference to the W3C DOM.

The following pages list the objects made available by the W3C DOM.

DOM CORE OBJECTS

This section describes and lists objects defined by the DOM standards—starting with the lowest level of DOM objects. All objects are in alphabetical order.

Low-Level DOM Objects

The DOM specification describes the `Node`, `NodeList`, and `NamedNodeMap` objects. These are the lowest-level objects in the DOM, and are the primary building blocks of higher-level objects.

Node

Defined in DOM Level 1, the `Node` object is the primary data type for the entire DOM. All objects in the DOM inherit from `Node`. There are 12 different types of `Node` objects; each type has an associated integer value. The following tables list the `Node` object's type values, properties, and methods.

Node Types

TYPE NAME	INTEGER VALUE	INTRODUCED	ASSOCIATED DATA TYPE
`ELEMENT_NODE`	1	Level 1	`Element`
`ATTRIBUTE_NODE`	2	Level 1	`Attr`
`TEXT_NODE`	3	Level 1	`Text`
`CDATA_SECTION_NODE`	4	Level 1	`CDATASection`
`ENTITY_REFERENCE_NODE`	5	Level 1	`EntityReference`
`ENTITY_NODE`	6	Level 1	`Entity`
`PROCESSING_INSTRUCTION_NODE`	7	Level 1	`ProcessingInstruction`
`COMMENT_NODE`	8	Level 1	`Comment`
`DOCUMENT_NODE`	9	Level 1	`Document`
`DOCUMENT_TYPE_NODE`	10	Level 1	`DocumentType`
`DOCUMENT_FRAGMENT_NODE`	11	Level 1	`DocumentFragment`
`NOTATION_NODE`	12	Level 1	`Notation`

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`attributes`	A `NamedNodeMap` containing the attributes of this node if it is an `Element`, or `null` otherwise.	Level 1
`childNodes`	A `NodeList` containing all children of this node.	Level 1
`firstChild`	Gets the first child of this node. Returns `null` if no child exists.	Level 1
`lastChild`	Gets the last child of this node. Returns `null` if no child exists.	Level 1
`localName`	Returns the local part of the node's qualified name (the part after the colon of the qualified name when namespaces are used). Used primarily in XML DOMs.	Level 2
`namespaceURI`	The namespace URI of the node, or `null` if not specified.	Level 2

`nextSibling`	Gets the node immediately following this node. Returns `null` if no following sibling exists.	Level 1
`nodeName`	Gets the name of this node.	Level 1
`nodeType`	An integer representing the type of this node. See previous table.	Level 1
`nodeValue`	Gets the value of this node, depending on the type.	Level 1
`ownerDocument`	Gets the `Document` object this node is contained in. If this node is a `Document` node, it returns `null`.	Level 1
`parentNode`	Gets the parent node of this node. Returns `null` for nodes that are currently not in the DOM tree.	Level 1
`prefix`	Returns the namespace prefix of this node, or `null` if not specified.	Level 2
`previousSibling`	Gets the node immediately before this node. Returns `null` if no previous sibling.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`appendChild(newChild)`	Adds the `newChild` to the end of the list of children.	Level 1
`cloneNode(deep)`	Returns a duplicate of the node. The returned node has no parent. If `deep` is `true`, this clones all nodes contained within the node.	Level 1
`hasAttributes()`	Returns a boolean value based on if the node has any attributes (if the node is an element).	Level 2
`hasChildNodes()`	Returns a boolean value based on whether the node has any child nodes.	Level 1
`insertBefore(newChild, refChild)`	Inserts the `newChild` node before the existing child referenced by `refChild`. If `refChild` is `null`, `newChild` is added at the end of the list of children.	Level 1
`removeChild(oldChild)`	Removes the specified child node and returns it.	Level 1
`replaceChild(newChild, oldChild)`	Replaces `oldChild` with `newChild` and returns `oldChild`.	Level 1

NodeList

The NodeList object is an ordered collection of nodes. The items contained in the NodeList are accessible via an index starting from 0.

A NodeList is a live snapshot of nodes. Any changes made to the nodes within the DOM are immediately reflected in every reference of the NodeList.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
length	The number of nodes in the list.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
item(index)	Returns the item at the specified index. Returns null if the index is greater than or equal to the list's length.	Level 1

NamedNodeMap

Objects referred to as NamedNodeMaps represent collections of nodes that can be accessed by name. This object does not inherit from NodeList. An element's attribute list is an example of a NamedNodeMap.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
length	The number of nodes in the map.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
getNamedItem(name)	Retrieves a node by the specified name.	Level 1
removeNamedItem(name)	Removes an item by the specified name.	Level 1
setNamedItem(node)	Adds a node to the list by using its nodeName property as its key.	Level 1

High-Level DOM Objects

These objects inherit Node and are the basis for even higher-level DOM objects as specified by the HTML DOM. These objects mirror the different node types.

The following objects are listed in alphabetical order. The `CDATASection`, `Comment`, `DocumentType`, `Entity`, `EntityReference`, `Notation`, and `ProcessingInstruction` objects are purposefully omitted from this section.

Attr

The `Attr` object represents an `Element` object's attribute. Even though `Attr` objects inherit from `Node`, they are not considered children of the element they describe, and thus are not part of the DOM tree. The `Node` properties of `parentNode`, `previousSibling`, and `nextSibling` return `null` for `Attr` objects.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`ownerElement`	Returns the `Element` object the attribute is attached to.	Level 2
`name`	Returns the name of the attribute.	Level 1
`value`	Returns the value of the attribute.	Level 1

Document

The `Document` object represents the entire HTML or XML document. It is the root of the document tree. The `Document` is the container for all nodes within the document, and each `Node` object's `ownerDocument` property points to the `Document`.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`docType`	The `DocType` object associated with this document. Returns `null` for HTML and XML documents without a document type declaration.	Level 1
`documentElement`	Returns the root element of the document. For HTML documents, the `documentElement` is the `<html/>` element.	Level 1
`implementation`	The `DOMImplementation` object associated with the `Document`.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`createAttribute(name)`	Returns a new `Attr` object with the specified name.	Level 1
`createAttributeNS(namespaceURI, qualifiedName)`	Returns an attribute with the given qualified name and namespace URI. Not for HTML DOMs.	Level 2

continues

(continued)

METHOD NAME	DESCRIPTION	INTRODUCED
`createComment(data)`	Returns a new `Comment` object with the specified data.	Level 1
`createCDATASection(data)`	Returns a new `CDATASection` object whose value is the specified data.	Level 1
`createDocumentFragment()`	Returns an empty `DocumentFragment` object.	Level 1
`createElement(tagName)`	Returns a new `Element` object with the specified tag name.	Level 1
`createElementNS(namespaceURI, qualifiedName)`	Returns an element of the specified qualified name and namespace URI. Not for HTML DOMs.	Level 2
`createTextNode(text)`	Returns a new `Text` object containing the specified text.	Level 1
`getElementById(elementId)`	Returns the `Element` with the specified ID value. Returns `null` if the element does not exist.	Level 2
`getElementsByTagName(tagName)`	Returns a `NodeList` of all `Element` objects with the specified tag name in the order in which they appear in the DOM tree.	Level 1
`getElementsByTagNameNS(namespaceURI, localName)`	Returns a `NodeList` of all elements with the specified local name and namespace URI. Elements returned are in the order they appear in the DOM.	Level 2
`importNode(importedNode, deep)`	Imports a node from another document. The source node is not altered or removed from its document. A copy of the source is created. If `deep` is `true`, all child nodes of the imported node are imported. If `false`, only the node is imported.	Level 2

DocumentFragment

The DocumentFragment object is a lightweight Document object. Its primary purpose is efficiency. Making many changes to the DOM tree, such as appending several nodes individually, is an expensive process. It is possible to append Node objects to a DocumentFragment object, which allows you to easily and efficiently insert all nodes contained within the DocumentFragment into the DOM tree.

The following code shows the use of a DocumentFragment:

```
var documentFragment = document.createDocumentFragment();

for (var i = 0; i < 1000; i++) {
    var element = document.createElement("div");
    var text = document.createTextNode("Here is test for div #" + i);

    element.setAttribute("id", i);

    documentFragment.appendChild(element);
}

document.body.appendChild(documentFragment);
```

Without the DocumentFragment object, this code would update the DOM tree 1,000 times, thus degrading performance. With the DocumentFragment object, the DOM tree is updated only once.

The DocumentFragment object inherits the Node object, and as such has Node's properties and methods. It does not have any other properties or methods.

Element

Elements are the majority of objects, other than text, that you will encounter in the DOM.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
tagName	Returns the name of the element. The same as Node.nodeName for this node type.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
getAttribute(name)	Retrieves the attribute's value by the specified name.	Level 1
getAttributeNS(namespaceURI, localName)	Returns the Attr object by local name and namespace URI. Not for HTML DOMs.	Level 2

continues

(continued)

METHOD NAME	DESCRIPTION	INTRODUCED
`getAttributeNode(name)`	Returns the `Attr` object associated with the specified name. Returns `null` if no attribute by that name exists.	Level 1
`getElementsByTagName(tagName)`	Returns a `NodeList` of all descendant elements with the specified `tagName` in the order in which they appear in the tree.	Level 1
`getElementsByTagNameNS(namespaceURI, localName)`	Returns a `NodeList` of all the descendant `Element` objects with the specified local name and namespace URI. Not for HTML DOMs.	Level 2
`hasAttribute(name)`	Returns a boolean value based on whether or not the element has an attribute with the specified name.	Level 2
`hasAttributeNS(namespaceURI, localName)`	Returns a boolean value based on whether the `Element` has an attribute with the given local name and namespace URI. Not for HTML DOMs.	Level 2
`querySelector(selector)`	Retrieves the first child element that matches the specified selector.	Level 3
`querySelectorAll(selector)`	Retrieves all child elements that match the specified selector.	Level 3
`removeAttribute(name)`	Removes the attribute with the specified name.	Level 1
`removeAttributeNS(namespaceURI, localName)`	Removes an attribute specified by the local name and namespace URI. Not for HTML DOMs.	Level 2
`removeAttributeNode(oldAttr)`	Removes and returns the specified attribute.	Level 1
`setAttribute(name, value)`	Creates and adds a new attribute, or changes the value of an existing attribute. The value is a simple string.	Level 1

`setAttributeNS(namespaceURI, qualifiedName, value)`	Creates and adds a new attribute with the specified namespace URI, qualified name, and value.	Level 2
`setAttributeNode(newAttr)`	Adds the specified attribute to the element. Replaces the existing attribute with the same name if it exists.	Level 1
`setAttributeNodeNS(newAttr)`	Adds the specified attribute to the element.	Level 2

Text

The `Text` object represents text content of an `Element` or `Attr` object.

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`splitText(indexOffset)`	Breaks the `Text` node into two nodes at the specified offset. The new nodes stay in the DOM tree as siblings.	Level 1

HTML DOM OBJECTS

In order to adequately interface with the DOM, the W3C extends the DOM Level 1 and 2 specifications to describe objects, properties, and methods, specific to HTML documents.

Most of the objects you'll interface with as a front-end developer are contained in this section.

Miscellaneous Objects: The HTML Collection

The `HTMLCollection` object is a list of nodes, much like `NodeList`. It does not inherit from `NodeList`, but `HTMLCollections` are considered live, like `NodeLists`, and are automatically updated when changes are made to the document.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`length`	Returns the number of elements in the collection.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
item(index)	Returns the element at the specified index. Returns null if index is larger than the collection's length.	Level 1
namedItem(name)	Returns the element using a name. It first searches for an element with a matching id attribute value. If none are found, it searches for elements with a matching name attribute value.	Level 1

HTML Document Objects: The HTML Document

The HTMLDocument object is the root of HTML documents and contains the entire content.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
anchors	Returns an HTMLCollection of all <a/> elements in the document that have a value assigned to their name attribute.	Level 1
applets	Returns an HTMLCollection of all <applet/> elements and <object/> elements that include applets in the document.	Level 1
body	Returns the element that contains the document's content. Returns the <body/> element, or the outermost <frameset/> element depending on the document.	Level 1
cookie	Returns the cookies associated with the document. Returns an empty string if none.	Level 1
domain	Returns the domain name of the server that served the document. Returns null if the domain name cannot be identified.	Level 1
forms	Returns an HTMLCollection of all <form/> elements in the document.	Level 1
images	Returns an HTMLCollection object containing all <img/> elements in the document.	Level 1
links	Returns an HTMLCollection of all <area/> and <a/> elements (with an href value) in the document.	Level 1

`referrer`	Returns the URL of the page that linked to the page. Returns an empty string if the user navigated directly to the page.	Level 1
`title`	The title of the document as specified by the `<title/>` element in the document's `<head/>` element.	Level 1
`URL`	The complete URL of the document.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`close()`	Closes a document.	Level 1
`getElementById(elementId)`	Returns the element with the given `elementId` or `null` if no element could be found. Removed in DOM Level 2 and added to the `Document` object.	Level 1
`getElementsByName(name)`	Returns an `HTMLCollection` of elements with the specified `name` attribute value.	Level 1
`open()`	Opens a document for writing.	Level 1
`write()`	Writes a string of text to the document.	Level 1
`writeln()`	Writes a string of text to the document followed by a newline.	Level 1

HTML Element Objects

HTML element attributes are exposed as properties of the various HTML element objects. Their data type is determined by the attribute's type in the HTML 4.0 specification.

Other than `HTMLElement`, all HTML element objects are described here in alphabetical order. The following pages do not contain a complete list of HTML element object types. Instead, only the following element object types are listed:

- `HTMLAnchorElement`
- `HTMLBodyElement`
- `HTMLButtonElement`
- `HTMLDivElement`
- `HTMLFormElement`
- `HTMLFrameElement`

- `HTMLFrameSetElement`
- `HTMLIFrameElement`
- `HTMLImageElement`
- `HTMLInputElement`
- `HTMLOptionElement`
- `HTMLParagraphElement`
- `HTMLSelectElement`
- `HTMLTableCellElement`
- `HTMLTableElement`
- `HTMLTableRowElement`
- `HTMLTableSectionElement`
- `HTMLTextAreaElement`

HTMLElement

`HTMLElement` is the base object for all HTML elements, much like how `Node` is the base object for all DOM nodes. Therefore, all HTML elements have the following properties.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`className`	Gets or sets the value of the element's `class` attribute.	Level 1
`id`	Gets or sets the value of the element's `id` attribute.	Level 1

HTMLAnchorElement

Represents the HTML `<a/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`accessKey`	Gets or sets the value of the `accessKey` attribute.	Level 1
`href`	Gets or sets the value of the `href` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`target`	Gets or set the value of the `target` attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`blur()`	Removes the keyboard focus from the element.	Level 1
`focus()`	Gives keyboard focus to the element.	Level 1

HTMLBodyElement

Represents the `<body/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`aLink`	Deprecated. Gets or sets the value of the `alink` attribute.	Level 1
`background`	Deprecated. Gets or sets the value of the `background` attribute.	Level 1
`bgColor`	Deprecated. Gets or sets the value of the `bgColor` attribute.	Level 1
`link`	Deprecated. Gets or sets the value of the `link` attribute.	Level 1
`text`	Deprecated. Gets or sets the value of the `text` attribute.	Level 1
`vLink`	Deprecated. Gets or sets the value of the `vlink` attribute.	Level 1

HTMLButtonElement

Represents `<button/>` elements.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`accessKey`	Gets or sets the value of the `accessKey` attribute.	Level 1
`disabled`	Gets or sets the value of the `disabled` attribute.	Level 1
`form`	Gets the `HTMLFormElement` object containing the button. Returns `null` if the button is not inside a form.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`type`	Gets the value of the `type` attribute.	Level 1
`value`	Gets or sets the value of the `value` attribute.	Level 1

HTMLDivElement

Represents the `<div/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`align`	Deprecated. Gets or sets the value of the `align` attribute.	Level 1

HTMLFormElement

Represents the `<form/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`action`	Gets or sets the value of the `action` attribute.	Level 1
`elements`	Returns an `HTMLCollection` object containing all form control elements in the form.	Level 1
`enctype`	Gets or sets the value of the `enctype` attribute.	Level 1
`length`	Returns the number of form controls within the form.	Level 1
`method`	Gets or sets the value of the `method` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`target`	Gets or sets the value of the `target` attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`reset()`	Resets all form control elements contained within the form to their default values.	Level 1
`submit()`	Submits the form. Does not fire the `submit` event.	Level 1

HTMLFrameElement

Represents the `<frame/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
contentDocument	Gets the Document object for the frame. Returns null if one isn't available.	Level 2
frameBorder	Gets or sets the value of the frameBorder attribute.	Level 1
marginHeight	Gets or sets the value of the marginHeight attribute.	Level 1
marginWidth	Gets or sets the value of the marginWidth attribute.	Level 1
name	Gets or sets the value of the name attribute.	Level 1
noResize	Gets or sets the value of the noResize attribute.	Level 1
scrolling	Gets or sets the value of the scrolling attribute.	Level 1
src	Gets or sets the value of the src attribute.	Level 1

HTMLFrameSetElement

Represents the <frameset/> element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
cols	Gets or sets the value of the cols attribute.	Level 1
rows	Gets or sets the value of the rows attribute.	Level 1

HTMLIFrameElement

Represents the <iframe/> element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
align	Deprecated. Gets or sets the value of the align attribute.	Level 1
contentDocument	Gets the Document object of the frame. Returns null if one doesn't exist.	Level 2

continues

(continued)

PROPERTY NAME	DESCRIPTION	INTRODUCED
`frameBorder`	Gets or sets the value of the `frameBorder` attribute.	Level 1
`height`	Gets or sets the value of the `height` attribute.	Level 1
`marginHeight`	Gets or sets the value of the `marginHeight` attribute.	Level 1
`marginWidth`	Gets or sets the value of the `marginWidth` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`noResize`	Gets or sets the value of the `noResize` attribute.	Level 1
`scrolling`	Gets or sets the value of the `scrolling` attribute.	Level 1
`src`	Gets or sets the value of the `src` attribute.	Level 1
`width`	Gets or sets the value of the `width` attribute.	Level 1

HTMLImageElement

Represents the `<img/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`align`	Deprecated. Gets or sets the value of the `align` attribute.	Level 1
`alt`	Gets or sets the value of the `alt` attribute.	Level 1
`border`	Deprecated. Gets or sets the value of the `border` attribute.	Level 1
`height`	Gets or sets the value of the `height` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`src`	Gets or sets the value of the `src` attribute.	Level 1
`width`	Gets or sets the value of the `width` attribute.	Level 1

HTMLInputElement

Represents the `<input/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`accessKey`	Gets or sets the value of the `accessKey` attribute.	Level 1
`align`	Deprecated. Gets or sets the value of the `align` attribute.	Level 1
`alt`	Gets or sets the value of the `alt` attribute.	Level 1
`checked`	Used when `type` is `checkbox` or `radio`. Returns a boolean value depending on whether or not the checkbox or radio button is checked.	Level 1
`defaultChecked`	Used when `type` is `checkbox` or `radio`. Gets or sets the checked attribute. The value does not change when other checkboxes or radio buttons are checked.	Level 1
`disabled`	Gets or sets the value of the `disabled` attribute.	Level 1
`form`	Gets the `HTMLFormElement` object containing the `<input/>` element. Returns `null` if the element is not inside a form.	Level 1
`maxLength`	Gets or sets the value of the `maxLength` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`readOnly`	Used only if `type` is `text` or `password`. Gets or sets the value of the `readonly` attribute.	Level 1
`size`	Gets or sets the value of the `size` attribute.	Level 1
`src`	If `type` is `image`, this gets or sets the value of the `src` attribute.	Level 1
`type`	Gets the value of the `type` attribute.	Level 1
`value`	Gets or sets the value of the `value` attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`blur()`	Removes keyboard focus from the element.	Level 1
`click()`	Simulates a mouse click for `<input/>` elements with `type button`, `checkbox`, `radio`, `reset`, and `submit`.	Level 1
`focus()`	Gives keyboard focus to the element.	Level 1
`select()`	Selects content of `<input/>` elements with `type text`, `password`, and `file`.	Level 1

HTMLOptionElement

Represents the `<option/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`defaultSelected`	Gets or sets the `selected` attribute. The value of this property does not change as other `<option/>` elements in the `<select/>` element are selected.	Level 1
`disabled`	Gets or sets the value of the `disabled` attribute.	Level 1
`form`	Gets the `HTMLFormElement` object containing the `<option/>` element. Returns `null` if the element is not inside a form.	Level 1
`index`	Gets the index position of the `<option/>` element in its containing `<select/>` element. Starts at `0`.	Level 1
`label`	Gets or sets the value of the `label` attribute.	Level 1
`selected`	Returns a boolean value depending on whether or not the `<option/>` element is currently selected.	Level 1
`text`	Gets the text contained within the `<option/>` element.	Level 1
`value`	Gets or sets the value of the `value` attribute.	Level 1

HTMLOptionCollection

The HTMLOptionCollection object was introduced in DOM Level 2. It contains a list of <option/> elements.

PROPERTY NAME	DESCRIPTION	INTRODUCED
length	Gets the number of <option/> elements in the list.	Level 2

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
item(index)	Retrieves the <option/> element at the specified index.	Level 2
namedItem(name)	Retrieves the <option/> element by the specified name. It first attempts to find an <option/> element with the specified id. If none can be found, it looks for <option/> elements with the specified name attribute.	Level 2

HTMLParagraphElement

Represents the <p/> element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
align	Deprecated. Gets or sets the value of the align attribute.	Level 1

HTMLSelectElement

Represents the <select/> element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
disabled	Gets or sets the value of the disabled attribute.	Level 1
form	Gets the HTMLFormElement object containing the <select/> element. Returns null if the element is not inside a form.	Level 1

continues

(continued)

PROPERTY NAME	DESCRIPTION	INTRODUCED
`length`	Returns the number of `<option/>` elements.	Level 1
`multiple`	Gets or sets the value of the `multiple` attribute.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`options`	Returns an `HTMLOptionsCollection` object containing the list of the `<option/>` elements.	Level 1
`selectedIndex`	Returns the index of the currently selected `<option/>` element. Returns `-1` if nothing is selected and returns the first `<option/>` element selected if multiple items are selected.	Level 1
`size`	Gets or sets the value of the `size` attribute.	Level 1
`type`	Gets the value of the `type` attribute.	Level 1
`value`	Gets or sets the current form control's value.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`add(element[, before])`	Adds an `<option/>` element to the `<select/>` element. If `before` is `null`, then `element` is added at the end of the list.	Level 1
`blur()`	Removes keyboard focus from the elements.	Level 1
`focus()`	Gives keyboard focus to the element.	Level 1
`remove(index)`	Removes the `<option/>` element at the given index. Does nothing if `index` is out of range.	Level 1

HTMLTableCellElement

Represents the `<td/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`align`	Deprecated. Gets or sets the value of the `align` attribute.	Level 1
`bgColor`	Deprecated. Gets or sets the value of the `bgcolor` attribute.	Level 1

cellIndex	The index of the cell in the row in DOM tree order.	Level 1
colSpan	Gets or sets the value of the colspan attribute.	Level 1
height	Deprecated. Gets or sets the value of the height attribute.	Level 1
noWrap	Deprecated. Gets or sets the value of the nowrap attribute.	Level 1
rowSpan	Gets or sets the value of the rowSpan attribute.	Level 1
vAlign	Gets or sets the value of the valign attribute.	Level 1
width	Deprecated. Gets or sets the value of the width attribute.	Level 1

HTMLTableElement

Represents the <table/> element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
align	Deprecated. Gets or sets the value of the align attribute.	Level 1
bgColor	Deprecated. Gets or sets the value of the bgcolor attribute.	Level 1
border	Gets or sets the value of the border attribute.	Level 1
cellPadding	Gets or sets the value of the cellPadding attribute.	Level 1
cellSpacing	Gets or sets the value of the cellSpacing attribute.	Level 1
rows	Returns an HTMLCollection containing all rows in the table.	Level 1
tBodies	Returns an HTMLCollection of the defined <tbody/> element objects in the table.	Level 1
tFoot	Returns the table's <tfoot/> element object (HTMLTableSectionElement), or null if one doesn't exist.	Level 1

continues

(continued)

PROPERTY NAME	DESCRIPTION	INTRODUCED
`tHead`	Returns the table's `<thead/>` element object (`HTMLTableSectionElement`), or `null` if one doesn't exist.	Level 1
`width`	Gets or sets the value of the `width` attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`createTFoot()`	Creates and returns a `<tfoot/>` element if one does not exist. Returns the existing `<tfoot/>` element if it exists.	Level 1
`createTHead()`	Creates and returns a `<thead/>` element if one does not exist. Returns the existing `<thead/>` element if it exists.	Level 1
`deleteRow(index)`	Deletes the row at the specified index.	Level 1
`deleteTFoot()`	Deletes the table's footer if one exists.	Level 1
`deleteTHead()`	Deletes the table's header if one exists.	Level 1
`insertRow(index)`	Inserts and returns a new row at the specified index. If `index` is `-1` or equal to the number of rows, the new row is appended to the end of the row list.	Level 1

HTMLTableRowElement

Represents the `<tr/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`align`	Deprecated. Gets or sets the value of the `align` attribute.	Level 1
`bgColor`	Deprecated. Gets or sets the value of the `bgcolor` attribute.	Level 1
`cells`	Returns an `HTMLCollection` containing the cells in the row.	Level 1

rowIndex	The index of the row in the table.	Level 1
sectionRowIndex	The index of the row relative to the section it belongs to (<thead/>, <tfoot/>, or <tbody/>).	Level 1
vAlign	Gets or sets the value of the valign attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
deleteCell(index)	Deletes the cell at the specified index.	Level 1
insertCell(index)	Inserts and returns an empty <td/> element. If index is -1 or equal to the number of cells in the row, the new cell is appended to the end of the list.	Level 1

HTMLTableSectionElement

Represents the <thead/>, <tbody/>, and <tfoot/> elements.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
align	Deprecated. Gets or sets the value of the align attribute.	Level 1
rows	Returns an HTMLCollection containing the rows of the section.	Level 1
vAlign	Gets or sets the value of the valign attribute.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
deleteRow(index)	Deletes the row at the specified index relative to the section.	Level 1
insertRow(index)	Inserts and returns a new row into the section at the specified index (relative to the section). If index is -1 or equal to the number of rows, the row is appended to the end of the list.	Level 1

HTMLTextAreaElement

Represents the `<textarea/>` element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`accessKey`	Gets or sets the value of the `accessKey` attribute.	Level 1
`cols`	Gets or sets the value of the `cols` attribute.	Level 1
`defaultValue`	Gets or sets the contents of the element. The value does not change when the content changes.	Level 1
`disabled`	Gets or sets the value of the `disabled` attribute.	Level 1
`form`	Gets the `HTMLFormElement` object containing the `<textarea/>` element. Returns `null` if the element is not inside a form.	Level 1
`name`	Gets or sets the value of the `name` attribute.	Level 1
`readOnly`	Used only if `type` is `text` or `password`. Gets or sets the value of the `readonly` attribute.	Level 1
`rows`	Gets or sets the value of the `rows` attribute.	Level 1
`type`	Gets the value of the `type` attribute. Always set to `textarea`.	Level 1
`value`	Gets or sets the current value of the element.	Level 1

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`blur()`	Removes keyboard focus from the element.	Level 1
`focus()`	Gives keyboard focus to the element.	Level 1
`select()`	Selects the contents of the element.	Level 1

HTML Media Objects

The `HTMLMediaElement` object is the base type for both `<video/>` and `<audio/>` elements.

HTMLMediaElement

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`autoplay`	Reflects the autoplay attribute, and determines where playback should automatically begin as soon as enough media is available.	HTML5
`buffered`	Gets the ranges of the media source that the browser has buffered.	HTML5
`controller`	Gets or sets the media controller associated with the element; returns `null` if none is linked.	HTML5
`controls`	Gets or sets the `controls` attribute, determining if the browser's default controls are displayed.	HTML5
`currentSrc`	Gets the absolute URL of the media.	HTML5
`currentTime`	The current playback time in seconds. Setting seeks the media to the specified time.	HTML5
`defaultMuted`	Gets or sets the `muted` attribute. This does not affect the audio after playback starts. Use the `muted` property for that.	HTML5
`defaultPlaybackRate`	The speed of playback. `1.0` is normal.	HTML5
`duration`	Gets the length of the media in seconds.	HTML5
`ended`	Indicates whether the media element has ended playback.	HTML5
`error`	The most recent error; `null` if no error has occurred.	HTML5
`loop`	Gets or sets the `loop` attribute; indicates whether the media should start over when it reaches the end.	HTML5
`mediaGroup`	Gets or sets the `mediagroup` attribute.	HTML5
`muted`	Mutes or unmutes the audio.	HTML5
`networkState`	The current state of fetching the media over the network.	HTML5
`paused`	Indicates whether the media element is paused.	HTML5

continues

(continued)

PROPERTY NAME	DESCRIPTION	INTRODUCED
`playbackRate`	Gets or sets the current playback rate.	HTML5
`played`	Gets the ranges that the media source has played, if any.	HTML5
`preload`	Gets or sets the `preload` attribute.	HTML5
`readyState`	Gets the readiness of the media.	HTML5
`seekable`	Gets the time ranges that the user can seek.	HTML5
`seeking`	Indicates whether the media is in the process of seeking to a new position.	HTML5
`src`	Gets or sets the `src` attribute.	HTML5
`volume`	Gets or sets the volume of the audio. 0.0 (silent) to 1.0 (loudest)	HTML5

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`canPlayType()`	Determines the likelihood the browser can play the given media type.	HTML5
`load()`	Begins loading the media content from the server.	HTML5
`pause()`	Pauses the media playback.	HTML5
`play()`	Begins or resumes the media playback.	HTML5

HTMLAudioElement

The `<audio/>` element does not have any unique properties or methods from `HTMLMediaElement`.

HTMLVideoElement

The `<video/>` element has a few unique properties.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`height`	Gets or sets the `height` attribute, determining the size of the display area.	HTML5

`poster`	Gets or sets the `poster` attribute, specifying the image to show while no video data is available.	HTML5
`videoHeight`	Gets the intrinsic height of the resource in CSS pixels.	HTML5
`videoWidth`	Gets the intrinsic width of the resource in CSS pixels.	HTML5
`width`	Gets or sets the `width` attribute, determine the size of the display area.	HTML5

DOM EVENT MODEL AND OBJECTS

The DOM event model was introduced in DOM Level 2. It describes an event system where every event has an event target. When an event reaches an event target, all registered event handlers on the event target are triggered for that specific event. The following objects are described by the DOM event model.

EventTarget

The `EventTarget` object is inherited by all `HTMLElement` objects in the DOM. This object provides the means for the registration and removal of event handlers on the event target.

Methods

METHOD NAME	DESCRIPTION
`addEventListener(type, listener, useCapture)`	Registers an event handler on an element. `type` is the event type to listen for, `listener` is the JavaScript function to call when the event is fired, and `useCapture` determines whether the event is captured or bubbles.
`removeEventListener(type, listener, useCapture)`	Removes a listener from the element.

Event

When an event fires, an `Event` object is passed to the event handler if one is specified. This object contains contextual information about an event.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`bubbles`	Indicates whether or not the event is a bubbling event.	Level 2
`cancelable`	Indicates whether or not the event can have its default action prevented.	Level 2
`currentTarget`	Indicates the `EventTarget` whose listeners are currently being processed.	Level 2
`target`	Indicates the `EventTarget` object to which the event was originally fired.	Level 2
`timeStamp`	Specifies the time (in milliseconds) at which the event was fired.	Level 2
`type`	The name of the event (remember: this is the name without the `on` prefix).	Level 2

Methods

METHOD NAME	DESCRIPTION	INTRODUCED
`preventDefault()`	Cancels the event, preventing the default action from taking place, only if the event is cancelable.	Level 2
`stopPropagation()`	Prevents further propagation of an event.	Level 2

MouseEvent

The `MouseEvent` object provides specific information associated with mouse events. `MouseEvent` objects contain not only the following properties, but also the properties and methods of the `Event` object.

Valid mouse events are shown in the following table.

EVENT NAME	DESCRIPTION
`click`	Occurs when the mouse button is clicked over an element. A `click` is defined as a `mousedown` and `mouseup` over the same screen location.
`mousedown`	Occurs when the mouse button is pressed over an element.
`mouseup`	Occurs when the mouse button is released over an element.
`mouseover`	Occurs when the mouse pointer moves onto an element.
`mousemove`	Occurs when the mouse pointer moves while it is over the element.
`mouseout`	Occurs when the mouse pointer moves away from an element.

Properties

PROPERTY NAME	DESCRIPTION	INTRODUCED
`altKey`	Returns a boolean value indicating whether or not the Alt key was pressed during the event's firing.	Level 2
`button`	Indicates which mouse button was pressed, if applicable. The number `0` represents the left button, `1` indicates the middle button, and `2` indicates the right button. Left-hand-configured mice reverse the buttons (right is `0`, middle is `1`, and left is `2`).	Level 2
`clientX`	The horizontal coordinate relative to the client area.	Level 2
`clientY`	The vertical coordinate relative to the client area.	Level 2
`ctrlKey`	Returns a boolean value indicating whether or not the Ctrl key was pressed when the event fired.	Level 2
`relatedTarget`	Indentifies a secondary `EventTarget`. Currently, this property is used with the `mouseover` event to indicate the `EventTarget` that the mouse pointer exited and with the `mouseout` event to indicate which `EventTarget` the pointer entered.	Level 2
`screenX`	The horizontal coordinate relative to the screen.	Level 2
`screenY`	The vertical coordinate relative to the screen.	Level 2
`shiftKey`	Returns a boolean value indicating whether or not the Shift key was pressed when the event fired.	Level 2

MISCELLANEOUS EVENTS

The following tables describe the events available in client-side JavaScript.

Mouse Events

EVENT	DESCRIPTION
`click`	Raised when the user clicks an HTML control.
`dblclick`	Raised when the user double-clicks an HTML control.
`mousedown`	Raised when the user presses a mouse button.

continues

(continued)

EVENT	DESCRIPTION
`mousemove`	Raised when the user moves the mouse pointer.
`mouseout`	Raised when the user moves the mouse pointer out from within an HTML control.
`mouseover`	Raised when the user moves the mouse pointer over an HTML control.
`mouseup`	Raised when the user releases the mouse button.

Keyboard Events

EVENT	DESCRIPTION
`keydown`	Raised when the user presses a key on the keyboard.
`keypress`	Raised when the user presses a key on the keyboard. This event will be raised continually until the user releases the key.
`keyup`	Raised when the user releases a key that had been pressed.

HTML Control Events

EVENT	DESCRIPTION
`blur`	Raised when an HTML control loses focus.
`change`	Raised when an HTML control loses focus and its value has changed.
`focus`	Raised when focus is set to the HTML control.
`reset`	Raised when the user resets a form.
`select`	Raised when the user selects text in an HTML control.
`submit`	Raised when the user submits a form.

Window Events

EVENT	DESCRIPTION
`load`	Raised when the window has completed loading.
`resize`	Raised when the user resizes the window.
`unload`	Executes JavaScript code when the user exits a document.

Media Events

EVENT	DESCRIPTION
`abort`	Raised when playback is aborted.
`canplay`	Sent when enough data is available that the media can be played.
`canplaythrough`	Fired when the entire media can be played without interruption.
`durationchange`	Raised when the metadata has changed.
`emptied`	Fires when the media has become empty.
`ended`	Sent when playback completes.
`error`	Sent when an error occurs.
`loadeddata`	The media's first frame has been loaded.
`loadedmetadata`	Fired when the media's metadata is loaded.
`loadstart`	Sent when downloading begins.
`pause`	Playback has been paused.
`play`	Playback begins after a pause.
`playing`	Raised when media begins to play.
`progress`	Indicates the progress of the media download.
`ratechange`	Fires when the playback rate changes.
`seeked`	Seeking has ended.
`seeking`	Playback is being moved to a new position.
`stalled`	Raised when the browser tries to download the media, but receives no data.
`suspend`	Sent when the loading of the media is suspended.
`timeupdate`	The media's `currentTime` has changed.
`volumechange`	Fires when the audio volume changes (both when volume is set and when muted).
`waiting`	Raised when playback is paused in order to download more data.

Other Events

EVENT	DESCRIPTION
`abort`	Raised when the user aborts loading an image.
`error`	Raised when an error occurs loading the page.