ARCHIVED 2010

SMPTE STANDARD

for Television — Declarative Data Essence — Document Object Model Level 0 (DOM-0) and Related Object Environment

1 Scope

This standard defines a document object model (DOM) and an object environment for use in manipulating HTML documents using the ECMAScript environment of declarative data essence. The standard reflects the best current practice for continuing use in television and other applications.

2 Reference and organization

2.1 Normative reference

The following standard contains provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standard indicated below.

ISO/IEC 16262:1998, Information Technology — ECMAScript Language Specification

2.2 Document organization

1 Scope
2 Reference and organization
3 Introduction
4 Definition (with IDL)
5 Event handling
6 Exception handling
7 Security
8 Standard color names
9 ECMAScript language binding
Annex A Object relationships
Annex B Bibliography

3 Introduction

This is a definition for a document object model and object environment (DOM-0), to allow orderly transition from existing current authoring practice to DOM-1 and DOM-2. DOM-0 is in widespread use today in the form of the objects and methods implemented by the popular browser vendors. This is an attempt to document something like DOM-0 (as was loosely defined by W3C). It provides an authoring standard to those who wish to make content that will be compatible with the largest number of *down level* browsers, such as those used in interactive television receivers supporting the declarative data essence content level 1 specification.

Standardized support for a DOM-0 is therefore required if one expects the existing tools, content, and author knowledge-base to be usable in the short term. And, DOM-0 support in browsers will continue to live on well beyond deployment of DOM-2, in order to avoid reauthoring of everything on the web.

4 Definition (with IDL)

This standard is intended to define the intersection of Microsoft Internet Explorer, version 3.0 (IE3) and Netscape Navigator, version 3.0 (NN3) relative to their DOM.

Additionally, there are some important features of both IE3 and NN3 that are not in ECMAScript, but not strictly part of the DOM either. These must also be addressed. So DOM-0 is a baseline DOM, as well as a set of miscellaneous functions including an extension to the ECMAScript String object, and the addition of some navigation objects. Collectively, this is referred to as DOM-0.

Copyright © 2002 by THE SOCIETY OF
MOTION PICTURE AND TELEVISION ENGINEERS
595 W. Hartsdale Ave., White Plains, NY 10607
(914) 761-1100

Approved
January 3, 2002

The definition is broken into three main groups:

– DOM
– ECMAScript Extensions
– Navigation

The DOM is the part that maps to a strict definition of a document object model (and is in theory easily mappable onto DOM-1). The ECMAScript Extensions are objects or methods that extend the definition of standard ECMAScript objects to handle current practice. Navigation is a set of objects that handle some of the common functions that have come to be expected in a given implementation of the client. These latter two categories are not DOM, but supporting objects that have become common for authors to use.

The strict DOM consists of the following objects:

– Anchor
– Button
– Checkbox
– Document
– Form
– Hidden
– Image
– Link
– Location
– Option
– Password
– Radio
– Reset
– Select
– Submit
– Text
– TextArea

The ECMAScript object extension is an extension of the standard ECMAScript:

– String

The new Navigation objects are:

– History
– Navigator
– Window

These are all defined as a flat object space. There is no inheritance or hierarchy and all methods are fully defined in each object definition.

The IDL for each object follows:

4.1 Anchor

```
Interface Anchor {
   attribute String name;
};
```

An HTML anchor tag object.

Properties
– Name - The HTML NAME attribute

Methods
– None

4.2 Button

```
Interface Button {
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void click( );
};
```

An HTML form button object.

Event handlers
– onBlur
– onClick
– onFocus

Properties
– Value - The string on the face of the button

Methods
– blur( )
– click( )

4.3 Checkbox

```
Interface Checkbox {
   attribute boolean checked;
   readonly attribute boolean defaultChecked;
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void click( );
   void focus( );
};
```

This is the HTML form checkbox object.

Event handlers
- onBlur
- onClick
- onFocus

Properties
- Checked - Boolean state of the checkbox (true if currently checked, else false)
- defaultChecked - The HTML CHECKED attribute
- Form - The Form containing this element
- Name - The HTML NAME attribute
- Type - The type of this form element
- Value - The HTML TYPE attribute

Methods
- blur( )
- click( )
- focus( )

4.4 Document

```
Interface AnchorSequence {
   Anchor item (in unsigned long index);
   readonly attribute unsigned long length;
} ;
Interface FormSequence {
   Form item (in unsigned long index);
   readonly attribute unsigned long length;
};
Interface ImageSequence {
   Image item (in unsigned long index);
   readonly attribute unsigned long length;
};
Interface LinkSequence {
   Link item (in unsigned long index);
   readonly attribute unsigned long length;
} ;
Interface Document {
   attribute String alinkcolor;
   readonly attribute AnchorSequence anchors;
   attribute String bgColor;
   attribute String cookie;
   attribute String fgColor;
   readonly attribute FormSequence forms;
   readonly attribute ImageSequence images;
   attribute String lastModified;
   attribute String linkColor;
   readonly attribute LinkSequence links;
   readonly attribute String location;
   readonly attribute String referrer;
   readonly attribute String title;
   readonly attribute String URL;
   attribute String vlinkColor;
   void clear( );
   void close( );
   void open (in String mimeType);
   void write (in String expr1, in String expr2, ...);
   void writeln (in String expr1, in String expr2, ...);
};
```

The document object.

Properties
- alinkColor - The HTML ALINK attribute. Note that the leading number is optional for all numeric values, and the Standard Color Names are supported.
- Anchors[ ] - An array of anchor objects in the order the anchor tags appear in the HTML document. Use anchors.length to get the number of anchors in a document.
- bgColor - The HTML BGCOLOR attribute. Note that the leading number is optional for all numeric values, and the Standard Color Names are supported.
- Cookie - All the cookie strings sent in the HTTP reply headers currently saved and concatenated. The syntax for each one is name=value; expires=expDate; the expDate format is: "Wdy, DD-Mmm-YY HH:MM:SS GMT".
- fgColor - The HTML FGCOLOR attribute. Note that the leading number is optional for all numeric values, and the Standard Color Names are supported.
- Forms[ ] - An array of form objects in the order the forms appear in the HTML file. Use forms.length to get the number of forms in a document. Note that only the formal array form is permitted (i.e., forms[index]).
- Images[ ] - An array of image objects in the order they appear in the HTML document. Use images.length to get the number of images in a document.
- lastModified - The value of the HTTP reply header field of the same name. Date formats are as permitted by HTTP/1.1.
- linkColor - The HTML LINK attribute. Note that the leading number is optional for all numeric values, and the Standard Color Names are supported.
- Links[ ] - An array of link objects in the order the hypertext links appear in the HTML document. Use links.length to get the number of links in a document.

- Location - A deprecated synonym for the URL property. This property exists only for compatibility with JavaScript 1.0.
- Referrer - String that contains the URL of the document that this was linked from.
- Title - The HTML TITLE tag.
- URL - A read-only string reflecting the actual URL of this document. Note that document.URL may differ from window.location because the actual URL of the document may be different from the URL used to request the document. Document.URL is not reflected in the window's location object.
- vlinkColor - The HTML VLINK attribute. Note that the leading number is optional for all numeric values, and the Standard Color Names are supported.

Methods
- Clear( ) - Clear the contents of this document.
- Close( ) - Close a previously opened new document and display it.
- Open( ) - Open a new document for creation.
  - mimeType - the ContentType of the document.
- Write( ) - Output one or more expressions to the currently open document.
  - Exprn - Any ECMAScript expression that results in a string.
- writeln( ) - Same as write( ) except a new line is inserted at the end of all the expressions.
  - Exprn - Any ECMAScript expression that results in a string.

4.5 Form

```
Interface ObjectSequence {
   Object item (in unsigned long index);
   readonly attribute unsigned long length;
} ;
Interface Form {
   attribute String action;
   readonly attribute ObjectSequence elements;
   attribute String encoding;
   readonly attribute long length;
   attribute String method;
   attribute String name;
   attribute String target;
   void submit ( );
};
```

The HTML Form object.

Event Handlers
- onSubmit

Properties
- Action - The HTML ACTION attribute.
- Elements[ ] - An array of objects for each form element in the order in which they appear in the form.
- Encoding - The string value of the MIME encoding as specified in ENCTYPE attribute.
- Length - Number of elements in *elements* above.
- Method - The HTML METHOD attribute.
- Name - The name of the form.
- Target - The HTML TARGET attribute.

Methods
- Submit( ) - Cause the form to be submitted.

4.6 Hidden

```
Interface Hidden {
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
};
```

The HTML Form Hidden field object.

Properties
- Form - The Form containing this element
- Name - The name of this form element
- Type - The type of this form element
- Value - The element value.

Methods
- <none>

4.7 History

```
Interface History {
   void back( );
   void forward( );
   void go (in long delta);
   void go (in String location);
};
```

This provides a global history list and management. Note that it is a property of the Window Object, and not per frame. Document URLs are added to the history whenever a link is taken by the user. Whenever one navigates *back* and then a peer branch is taken, the

other peer tree (if any) of document URLs is pruned from the history. Navigating *back* causes the document that the user navigated from to reach the current document to be displayed. Navigating *forward* causes the document most recently reached from the current document to be displayed. The history list includes the navigation from triggers just like they were links that the user had taken.

When matching with go(), the **location** argument is matched against URLs in the history list and a partial match occurs when the full string content of **location** matches the initial set of characters in one of the URLs (i.e., the match is anchored to the beginning of the URL). This is started at the current position in the forward direction, wrapping around to the top.

Properties
- <none>

Methods
- Back( ) - Load the previous URL in the history list.
- Forward( ) - Load the next URL in the history list.
- Go( ) - Load a specific URL in the history list.
  - Delta - An integer number of an item in the history list relative to the current document. A positive number indicates the equivalent of *delta* forward( ) calls. A negative number indicates the equivalent of *delta* back( ) calls.
  - Location - A URL of an item in the history list. Partial matches are permitted where the first partial string match is loaded.

4.8 Image

```
Interface Image {
   attribute long border;
   readonly attribute boolean complete;
   attribute long height;
   attribute long hspace;
   attribute String lowsrc;
   attribute String name;
   attribute String src;
   attribute long vspace;
   attribute long width;
};
```

The image object reflects an image included in an HTML document.

Event handlers
- onLoad

Properties
- Border - An integer value reflecting the width of the image's border in pixels.
- Complete - A boolean value indicating whether the image has finished loading.
- Height - An integer value reflecting the height of an image in pixels.
- hspace - An integer value reflecting the HSPACE attribute of the <IMG> tag, which controls the amount of white space, in pixels, to the left and right of the image.
- lowsrc - A string value containing the URL of the low-resolution version of the image to load.
- Name - A string value indicating the name of the image object.
- src - A string value indicating the URL of the image.
- vspace - An integer value reflecting the VSPACE attribute of the <IMG> tag, which controls the amount of white space, in pixels, above and below the image.
- Width - An integer value indicating the width of an image in pixels.

4.9 Link

```
Interface Link {
   attribute String hash;
   attribute String host;
   attribute String hostname;
   attribute String href;
   attribute String pathname;
   attribute String port;
   attribute String protocol;
   attribute String search;
   attribute String target;
};
```

The HTML Link object of the form:
protocol://host:port/pathname?search#hash

Event handlers
- onClick
- onMouseOut
- onMouseOver

Properties
- Hash - The URL fragment component.
- Host -The URL domain name (or IP address) part of the authority component.
- hostname - The entire authority component, including port number, if present.
- href - The entire URL.
- pathname - The URL path component.

– Port - The port number of the URL authority component; zero if none.
– Protocol - The scheme name, including the ":".
– Search - The URL query component.
– Target - The HTML TARGET attribute.

Methods
– <none>

4.10 Location

```
Interface Location {
   attribute String hash;
   attribute String host;
   attribute String hostname;
   attribute String href;
   attribute String pathname;
   attribute String port;
   attribute String protocol;
   attribute String search;
};
```

The location object of a window is a reference to Location object, which is a representation of the URL of the document currently being displayed in that window. Location represents the URL used to request the current document, but may not equal the actual URL of the document, which is stored in the string document.URL. In addition to its properties, the Location object can be used as if it were itself a primitive string value. If the value of a Location object is read, the string obtained is the same as that obtained if the href property of the object is read. Assigning a URL to the location property of a window causes the browser to load and display the contents of the URL assigned to it.

Properties
– Hash - The URL fragment component.
– Host - The URL domain name (or IP address) part of the authority component.
– hostname - The entire authority component, including port number, if present.
– href - The entire URL.
– pathname - The URL path component.
– Port - The port number of the URL authority component; zero if none.
– Protocol - The scheme name, including the ":".
– Search - The URL query component.

Methods
– <none>

4.11 Navigator

```
InterfaceNavigator {
   readonly attribute String appCodeName;
   readonly attribute String appName;
   readonly attribute String appVersion;
   readonly attribute String userAgent;
};
```

Properties
– appCodeName - The code name of the browser implementation.
– appName - The browser vendor.
– appVersion- The browser version number.
– userAgent - The HTTP Request Header UserAgent field value.

Methods
– <none>

4.12 Option

```
Interface Option {
   readonly attribute boolean defaultSelected;
   readonly attribute long index;
   attribute boolean selected;
   readonly attribute String text;
   attribute String value;
};
```

The HTML Form Option object.

Properties
– defaultSelected - Initial state of the option selection.
– Index - An integer value specifying the position of the option in the select list.
– Selected - The current selection state.
– Text - The text of an option in the selection list.
– Value - The value returned to the server.

Methods
– <none>

4.13 Password

```
Interface Password {
   readonly attribute String defaultValue;
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void focus( );
```

```
   void select( );
};
```

The HTML Form Password object.

Event handlers
- onBlur
- onFocus

Properties
- defaultValue - The HTML VALUE attribute.
- Form - The Form containing this element.
- Name - The name of this form element.
- Type - The HTML TYPE attribute.
- Value - The current value of the field.

Methods
- blur( )
- focus( )
- select( )

4.14 Radio

```
Interface Radio {
   attribute boolean checked;
   readonly attribute boolean defaultChecked;
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void click( );
   void focus( );
};
```

The HTML Form Radio object.

Event handlers
- onBlur
- onClick
- onFocus

Properties
- Checked - Boolean state of the radio button.
- defaultChecked - The HTML CHECKED attribute.
- Form - The form containing this element.
- Name - The HTML NAME attribute.
- Type - The type of this form element.
- Value - The HTML VALUE attribute.

Methods
- blur( )
- click( )
- focus()

4.15 Reset

```
Interface Reset {
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void click( );
   void focus( );
};
```

The HTML Form Reset button.

Event handlers
- onBlur
- onClick
- onFocus

Properties
- Form - The Form containing this element.
- Name - The name of this form element.
- Type - The type of this form element.
- Value - The HTML VALUE attribute.

Methods
- blur( )
- click( )
- focus( )

4.16 Select

```
Interface OptionSequence {
   Option item (in unsigned long index);
   readonly attribute unsigned long length;
};
Interface Select {
   readonly attribute Form form;
   readonly attribute long length;
   readonly attribute String name;
   readonly attribute OptionSequence options;
   attribute long selectedIndex;
   readonly attribute String type;
   void blur( );
   void focus( );
};
```

The HTML Form Select object.

Event handlers
- onBlur
- onChange
- onFocus

Properties
- Form - The Form that contains this element.
- Length - Number of options in the selection list.
- Name - The name of this form element.
- Options[] - The array of Option objects, reflecting each of the options in the selection list in the order they appear.
- selectedIndex - The index (position) of the currently selected option in the selection list.
- Type - The type of this form element.

Methods
- blur( )
- focus( )

4.17 String

```
Interface String: ECMAString {
   String anchor (in String nameAttribute);
   String big( );
   String blink( );
   String bold( );
   String fixed( );
   String fontcolor (in String color);
   String fontsize (in long size);
   String italics( );
   String link (in String hrefAttribute);
   String small( );
   String strike( );
   String sub( );
   String sup( );
};
```

This is an extension of the standard ECMAScript String object. The above IDL shows that this is derived from the ECMAScript object but, in reality, it is an extension of the standard object.

Properties
- <none>

(Extra) methods
- Anchor( ) - Returns the string "<A NAME=nameAttribute>string</A>".
  - nameAttribute - Value of the HTML NAME attribute.
- Big( ) - Returns the string "<BIG>string</BIG>".
- Blink( ) - Returns the string "<BLINK>string</BLINK>".
- Bold( ) - Returns the string "<B>string</B>"
- Fixed( ) - Returns the string "<TT>string</TT>".
- FontColor( ) - Returns the string "<FONT COLOR=color>string</FONT>".
  - Color - The value of the HTML COLOR attribute.
- Fontsize( ) - Returns the string "<FONTSIZE =size>string</FONTSIZE>".
  - Size - The HTML FONTSIZE tag value.
- Italics( ) - Returns the string "<I>string</I>".
- Link( ) - Returns the string "<A HREF=hrefAttribute>string</A>".
  - hrefAttribute - The value of the HTML HREF attribute.
- Small( ) - Returns the string "<SMALL>string</SMALL>".
- Strike( ) - Returns the string "<STRIKE>string</STRIKE>".
- Sub( ) - Returns the string "_{string}".
- Sup( ) - Returns the string "^{string}".

4.18 Submit

```
Interface Submit {
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void click( );
   void focus( );
};
```

The HTML Form Submit button.

Event handlers
- onBlur
- onClick
- onFocus

Properties
- Form - The Form that contains this element.
- Name - The name of this form element.
- Type - The type of this form element.
- Value - The HTML VALUE attribute.

Methods
- blur( )
- click( )
- focus( )

4.19 Text

```
Interface Text {
   readonly attribute String defaultValue;
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void focus( );
   void select( );
};
```

The HTML Form Text Object.

Event handlers
- onBlur
- onChange
- onFocus

Properties
- defaultValue - The HTML VALUE attribute.
- Form - The Form that contains this element.
- Name - The name of this form element.
- Type - The HTML TYPE attribute.
- Value - The current string value of the field.

Methods
- blur( )
- focus( )
- select( )

4.20 TextArea

```
Interface TextArea {
   readonly attribute String defaultValue;
   readonly attribute Form form;
   readonly attribute String name;
   readonly attribute String type;
   attribute String value;
   void blur( );
   void focus( );
   void select( );
};
```

The HTML Form TextArea object.

Event handlers
- onBlur
- onChange
- onFocus

Properties
- defaultValue - The HTML VALUE attribute.
- Form - The Form that contains this element.
- Name - The name of this element.
- Type - The HTML TYPE attribute.
- Value - The current value of the string field.

Methods
- blur( )
- focus( )
- select( )

4.21 Window

```
Interface WindowSequence {
   Window item (in unsigned long index);
   readonly attribute unsigned long length;
};
Interface Window {
   readonly attribute String defaultStatus;
   readonly attribute Document document;
   readonly attribute WindowSequence frames;
   readonly attribute History history;
   readonly attribute long length;
   attribute Location location;
   readonly attribute String name;
   readonly Navigator navigator;

   readonly Window opener;
   readonly attribute Window parent;
   readonly attribute Window self;
   attribute String status;
   readonly attribute Window top;
   readonly attribute Window window;
   void alert (in String message);
   void clearTimeout (in long timeoutID);
   void close ( );
   boolean confirm (in String message);
   void open (in String url, in String name, in String
features, in boolean replace);
   String prompt (in String message);
   String prompt (in String message, in String inputDefault);
   String prompt (in String message, in long inputDefault);
   long setTimeout (in String expr, in long msec);
};
```

This provides basic window management functions. Note that only one window is supported, so there are restrictions on properties and methods that imply or act on multiple windows. Authors should be aware that browser implementations may restrict access to Window object information and navigation control under some circumstances, such as from scripts running

within frames, in order to protect the privacy of user information, and maintain control by the host document/application.

Event handlers
- onLoad
- onUnload

Properties
- defaultStatus - The default string in the "status bar" (implementation dependent - see status below).
- Document - Document object of the document in this window.
- Frames[] - An array of window objects that represents the frames in this window. Frames appear in the array in the order in which they appear in the HTML source code. If there are no frames, then this array is empty, and the length value below is zero.
- History - The history object for URLs displayed in this window (note that this is a global history list).
- Length - Length of the frame array.
- Location - The location (URL) object for this window, representing the URL used to request the current document. Setting the value of window.location to a URL string sets the href value of the window's associated location object, and causes the window to navigate to the URL, if window. location is set to a new URL. Note that the requested URL in window.location may be different from the actual URL of the delivered document, which is stored in document.URL.
- Name - Name of this window (implementation specific).
- Navigator - The pointer to the navigator object.
- Opener - The URL string of the document that opened the window.
- Parent - The parent window (of a frame).
- Self - Pointer to this window object.
- Status - String to be displayed (perhaps transiently) in the status bar. This is implementation dependent, and cannot be counted on to be reliably displayed to the user. Its use is discouraged and is supported here for script compatibility.
- Top - Topmost window in the window/frame hierarchy.
- Window - Same as self.

Methods
- Alert( ) - Display an *alert* message (implementation dependent). Note that this may not block in event handler code, so its use should be avoided in there.
  - Message - The message string to display.
- clearTimeout() - Clear the timer set by setTimeout.
  - timeoutID - The ID of the timer.
- Close( ) - Close (and cause to render) the window.
- Confirm( ) - Display a *confirm* message (implementation dependent). Note that this may not block in event handler code, so its use should be avoided in there.
  - Message - The message string to display.
- Open( ) - Open a new window. The exact receiver behavior of this (single window or multiple window system, etc.) is implementation dependent.
  - url - The URL string of the document to open.
  - Name - String name of the window.
  - Features - String of features about the target window, of the form, "token[=value]". Any tokens are legal, but the ones a content author should expect to work are:
    - Height - Height in pixels
    - Status - Boolean to enable the status line
    - Width - Width in pixels
  - Replace - A boolean, if true, replaces the current window.
- Prompt( ) - Display a *prompt* message (implementation dependent) and return the user entry. Note that this may not block in event handler code, so its use should be avoided in there.
  - Message - The message string to display.
  - inputDefault - The value first displayed in the prompt area.
- setTimeout( ) - Evaluate an ECMAScript expression after a period of time.
  - expr - The expression to evaluate.
  - msec - The milliseconds to wait before evaluation.

5 Event handling

User interface events are handled in this model very differently from those proposed in DOM-2. These are done with a small set of methods to allow the simulation of events, and a small set of HTML attributes that are *handler-like*. The methods allow programmatic control over causing certain events, and the callback *handlers* are *evoked* (more on the quotes below) when the event occurs. There are no object attributes that represent these *handlers*. The ECMAScript code for these can only be defined by the HTML tags.

5.1 Methods

The methods are:

- blur( )
- click( )
- focus( )
- select( )

blur( ) – This removes focus from the object.

click( ) – This simulates a mouse click on the object.

focus( ) – This sets the focus on the object.

select( ) – This selects in a field in some input objects as if the user had used the mouse to highlight and select an item.

5.2 Handlers

The callback *handlers* are actually string values containing in-line simple or compound ECMAScript statements, and not actual functions. So these are never actually invoked, but are documented here as *functions* for conceptual clarity only.

5.2.1 Scope

The scope chain for executing an event handler begins with the object which emits the event and proceeds upwards through the object reference hierarchy as follows: {Link,Input*}, [Form], Document, Window (where Input* refers to any of the Form input element types).

Within the context of a global function, i.e., a function declared in a <script> element, *this* is bound to the applicable Window object. The applicable Window object is determined by the scope chain of the caller of such a function.

5.2.2 Handler context

Within the context of an event handler, *this* is bound to the object which emits the event.

5.2.3 Return values

The return value is that set using an ECMAScript *return* statement in the event handler compound statement.

5.2.4 Handler summary

The handler *functions* are:

- onBlur( )
- onChange( )
- onClick( )
- onFocus( )
- onLoad( )
- onMouseOut( )
- onMouseOver( )
- onSubmit( )
- onUnLoad( )

onBlur – This HTML attribute allows the user to set an ECMAScript compound statement to execute when focus is lost on the object; *this* is any Input object as appropriate. Any return value is ignored.

onChange – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the field value is changed by the user, and can be used to validate fields before submission; *this* is one of {Select, Text, TextArea} as appropriate. Any return value is ignored.

onClick – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the user clicks on the object; *this* is one of {Button, Checkbox, Link, Radio} as appropriate. A return value of *false* will cancel the default action, and *true* will perform the default action.

onFocus – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the object gets focus; *this* is any Input object as appropriate. Any return value is ignored.

onLoad – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the object (window, frame, or image) is fully loaded; *this* is the Window or Image object as appropriate. Any return value is ignored.

onMouseOut – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the mouse is moved out of the object's defined area; *this* is the Link object. Any return value is ignored.

onMouseOver – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the mouse is moved into the object's defined area; *this* is the Link object. A return value of *true* will prevent the system from displaying the URL in the status bar.

onSubmit – This HTML attribute allows the user to set an ECMAScript compound statement to execute when the user submits a form; *this* is the Form object. A return value of *false* will prevent the submission.

onUnload – This HTML attribute allows the user to set an ECMAScript compound statement to execute just

prior to when the document in this window is unloaded. Any return value is ignored.

6 Exception handling

This version of this standard only contemplates an ECMAScript, edition 2 binding. There is no exception handling in ECMAScript, edition 2. Therefore, exception handling, if any, is entirely implementation dependent.

7 Security

ECMAScript in a frame is allowed to access methods and properties in other frames whose URLs specify the same host name and port. Note that there may be security policies that restrict usage of the methods, but that is implementation dependent.

8 Standard color names

The supported color names are those defined in CSS1. The values for the colors shall be from the HTML 4 specification.

9 ECMAScript language binding

Note that this is subordinate to the IDL definitions above. If there is any inconsistency, then refer to the IDL.

Object Anchor
  Properties:
    String name

Object Button
  Properties:
    Form form (readonly)
    String name (readonly)
    String type (readonly)
    String value
  Methods:
    void blur( )
    void click( )

Object Checkbox
  Properties:
    Boolean checked
    Boolean defaultChecked (readonly)
    Form form (readonly)
    String name (readonly)
    String type (readonly)
    String value
  Methods:
    void blur( )
    void click( )
    void focus( )

Object Document
  Properties:
    String alinkColor
    Array anchors (readonly of type Anchor)
    String bgcolor
    String cookie
    String fgColor
    Array forms (readonly of type Form)
    Array images (readonly of type Image)
    String lastModified
    String linkColor
    Array links (readonly of type Link)
    String location (readonly)
    String referrer
    String title (readonly)
    String URL (readonly)
    String vlinkColor
  Methods:
    void clear( )
    void close( )
    void open (String mimeType)
    void write (String expr1, String expr2, ...)
    void writeln (String expr1, String expr2, ...)

Object Form
  Properties:
    String action
    Array elements (readonly of type Object)
    String encoding
    Number length (readonly)
    String method
    String name
    String target
  Methods:
    void submit( )

Object Hidden
  Properties:
    Form form (readonly)
    String name (readonly)
    String type (readonly)
    String value

Object History
  Methods:
    void back( )
    void forward( )

void go (Number delta)
void go (String location)

Object Image
Properties:
Number border
Boolean complete (readonly)
Number height
Number hspace
String lowsrc
String name
String src
Number vspace
Number width

Object Link
Properties:
String hash
String host
String hostname
String href
String pathname
String port
String protocol
String search
String target

Object Location
Properties:
String hash
String host
String hostname
String href
String pathname
String port
String protocol
String search

Object Navigator
Properties:
String appCodeName (readonly)
String appName (readonly)
String appVersion (readonly)
String userAgent (readonly)

Object Option
Properties:
Boolean defaultSelected (readonly)
Number index (readonly)
Boolean selected
String text (readonly)
String value

Object Password
Properties:
String defaultValue (readonly)
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void focus( )
void select( )

Object Radio
Properties:
Boolean checked
Boolean defaultChecked (readonly)
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void click( )
void focus( )

Object Reset
Properties:
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void click( )
void focus( )

Object Select
Properties:
Form form (readonly)
Number length (readonly)
String name (read only)
Array options (readonly of type option)
Number selectedIndex
String type (readonly)
Methods:
void blur( )
void focus( )

Object String (extensions to standard object - not derived)
Methods:
String anchor (String nameAttribute);
String big( );

String blink( );
String bold( );
String fixed( );
String fontcolor (String color);
String fontsize (Number size);
String italics( );
String link (String hrefAttribute);
String small( );
String strike( );
String sub( );
String sup( );

Object Submit
Properties:
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void click( )
void focus( )

Object Text
Properties:
String defaultValue (readonly)
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void focus( )
void select( )

Object TextArea
Properties:
String defaultValue (readonly)
Form form (readonly)
String name (readonly)
String type (readonly)
String value
Methods:
void blur( )
void focus( )
void select( )

Object Window
Properties:
String defaultStatus
Document document (readonly)
Array frames (readonly of type Window)
History history (readonly)
Number length (readonly)
Location location
String name (readonly)
Navigator navigator (readonly)
Window opener (readonly)
Window parent (readonly)
Window self (readonly)
String status
Window top (readonly)
Window window (readonly)
Methods:
void alert (String message)
void clearTimeout (Number timeoutID)
void close( )
Boolean confirm (String message)
open (String url, String name, String features,
boolean replace)
String prompt (String message, Number
inputDefault)
Number setTimeout (String expr, Number msec)

Annex A (informative)
Object relationships

Each of the objects in clause 4 is defined in the following clauses by means of an IDL Interface specification, one Interface specification per object. No object has an inheritance relationship with any other object. However, some of the objects, e.g., Window, Document, Select, have attributes that reference other object instances or sequences of other object instances. Figure A.1 shows how object instances are linked as a result of these references. The arrow means *has attribute which references*. The notation "<object-name>[ ]" is a reference to an object which is a sequence of <object-name>.

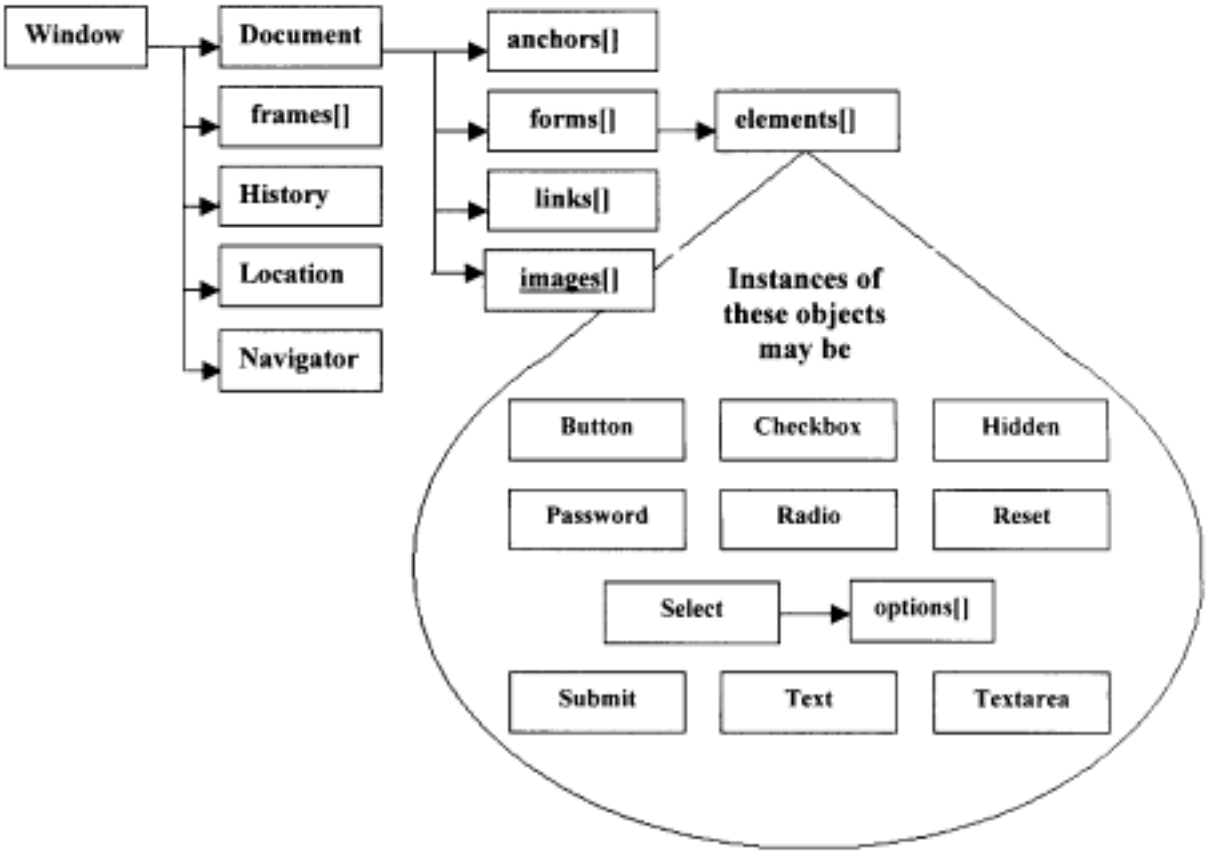

Figure A.1 – Linkage of object instances

Annex B (informative)
Bibliography

Microsoft, Javascript Host DOM Objects [product documentation]

Microsoft, JScript [product documentation]

Netscape, Javascript 1.3 [product documentation]

Object Management Group (OMG) CORBA/IIOP 2.3.1, The Common Object Request Broker: Architecture and Specification, Section 3, OMG IDL Syntax and Semantics

SMPTE 363M-2002, Television — Declarative Data Essence — Content Level 1

W3C Recommendation, Document Object Model (DOM) Level 1 Specification