
SMPTE STABLE DOCUMENT



The attached SMPTE Engineering Document has been declared “Stable” by the controlling Technology Committee.

The SMPTE Operations Manual for Standards states:

A document should be stabilized if it is believed to be substantially correct, does not contain harmful or misleading recommendations, may still be relevant to equipment or practices in use, is stable, but does not represent current technology, and need not be subject to future reviews.

A Stable document shall still be made available and offered for sale by the Society, but it shall be prefaced by a cover page explaining its current status.

At any time, a Technology Committee may revise, amend, or otherwise initiate a new Project on a Stable document.

A Stable document is “In Force”, and not deprecated or withdrawn.

*** * * * ***

Note:

SMPTE “Stable” documents were previously described as “Archived” and the attached document may be marked as “Archived”. The status of a SMPTE document described as “Archived” is exactly as described above for a “Stable” document.

Stable documents may not adhere to the latest style and format of SMPTE documents, or to current usage of normative language. Suitable care should be taken in interpretation.

SMPTE STANDARD

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



Page 1 of 15 pages

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.

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