Abstract

This non-normative reference describes the HTML markup language and provides details to help producers of HTML content create documents that conform to the language. It is intended to complement the normative conformance criteria defined in the HTML5: A vocabulary and associated APIs for HTML and XHTML specification, as well as information in related deliverables published by the HTML Working Group and from other sources. By design, this reference does not describe related APIs in detail, nor attempt to explain how implementations that are consumers of HTML content are meant to process documents (those areas are covered by the HTML5 specification itself), nor attempt to also be a tutorial or “how to” authoring guide.
This document is the 13 January 2011 Working Draft of HTML: The Markup Language Reference. If you’d like to comment on this document, the preferred means for commenting is to submit your comments through the HTML Working Group bugzilla database, with the Component field set to HTML5: The Markup Language. Alternatively, you can send comments by e-mail to public-html-comments@w3.org (archived).

This document was published by the W3C HTML Working Group, part of the HTML Activity in the W3C Interaction Domain.

Instability and incompleteness of this document

This document is a working draft — all parts of it remain subject to significant change or removal, and some parts are currently incomplete; in particular, many elements currently lack complete documentation in the following subsections:

- **Examples**
  The documentation for each element ideally have at least one conformant example and at least one non-conformant example.

- **Details**
  The purpose of this subsection is to provide, where needed, additional details about the semantics of the element and its attributes.

- **Additional constraints and admonitions**
  This purpose of this subsection is to provide, where needed, additional document-conformance constraints and usage admonitions for the element and its attributes (in addition to those already documented in the Permitted content, Permitted attributes, Permitted parent elements, and Tag omission subsections).

Publication as a Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. This document is informative only. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Table of Contents

1. Introduction
2. Terminology
3. Documents
4. HTML syntax
5. HTML elements organized by function
6. HTML elements
7. Common content models
8. Global attributes
9. Data types (common microsyntaxes)

A. References
B. Acknowledgments

Index of terms

Full Table of Contents

1. Introduction
2. Terminology
3. Documents
   3.1. The HTML language and HTML and XML syntaxes
   3.2. The HTML namespace and MIME types
   3.3. Conformant documents
   3.4. Case insensitivity in tag names and attribute names
4. HTML syntax
   4.1. The doctype
   4.2. Character encoding declaration
   4.3. Elements
   4.4. Attributes
   4.5. Text and character data
   4.6. Character references
   4.7. Comments
   4.8. SVG and MathML elements in HTML documents
   4.9. CDATA sections in SVG and MathML contents
5. HTML elements organized by function
6. HTML elements
   a – hyperlink  
   abbr – abbreviation
   address – contact information
   area – image-map hyperlink
   article – article  
   aside – tangential content  
   audio – audio stream
   b – offset text conventionally styled in bold  
   base – base URL
   bidi – BiDi isolate  
   bdo – BiDi override
   blockquote – block quotation
   body – document body
   br – line break
   button – button
   button type=submit – submit button
   button type=reset – reset button
   button type=button – button with no additional semantics
   canvas – canvas for dynamic graphics

caption – table title

cite – cited title of a work

code – code fragment

col – table column

colgroup – table column group

command – command

command type=command – command with an associated action

command type=radio – selection of one item from a list of items

command type=checkbox – state or option that can be toggled

datalist – predefined options for other controls

dd – description or value

del – deleted text

details – control for additional on-demand information

dfn – defining instance

div – generic flow container

dl – description list

dt – term or name

em – emphatic stress

embed – integration point for plugins

fieldset – set of related form controls

figcaption – figure caption

figure – figure with optional caption

footer – footer

form – user-submittable form

h1 – heading

h2 – heading

h3 – heading

h4 – heading

h5 – heading

h6 – heading

head – document metadata container

header – header

hgroup – heading group

hr – thematic break

html – root element

i – offset text conventionally styled in italic

iframe – nested browsing context (inline frame)

img – image

input – input control

input type=text – text-input field

input type=password – password-input field

input type=checkbox – checkbox

input type=radio – radio button

input type=button – button
input type=submit – submit button
input type=reset – reset button
input type=file – file upload control
input type=hidden – hidden input control
input type=image – image-coordinates input control
input type=datetime – global date-and-time input control NEW
input type=datetime-local – local date-and-time input control NEW
input type=date – date input control NEW
input type=month – year-and-month input control NEW
input type=time – time input control NEW
input type=week – year-and-week input control NEW
input type=number – number input control NEW
input type=range – imprecise number-input control NEW
input type=email – e-mail address input control NEW
input type=url – URL input control NEW
input type=search – search field NEW
input type=tel – telephone-number-input field NEW
input type=color – color-well control NEW
ins – inserted text
kbd – user input
keygen – key-pair generator/input control NEW
label – caption for a form control
legend – title or explanatory caption
li – list item
link – inter-document relationship metadata
map – image-map definition
mark – marked (highlighted) text NEW
menu – list of commands CHANGED
meta – metadata CHANGED
meta name – name-value metadata
meta http-equiv=refresh – “refresh” pragma directive
meta http-equiv=default-style – “preferred stylesheet” pragma directive
meta http-equiv=content-language – “default language” pragma directive OBSOLETE
meta charset – document character-encoding declaration NEW
meta http-equiv=content-type – document character-encoding declaration
meter – scalar gauge NEW
nav – group of navigational links NEW
noscript – fallback content for script
object – generic external content
ol – ordered list
optgroup – group of options
option – option
output – result of a calculation in a form NEW
p – paragraph
param – initialization parameters for plugins
pre – preformatted text
progress – progress indicator  NEW
q – quoted text
rp – ruby parenthesis  NEW
rt – ruby text  NEW
ruby – ruby annotation  NEW
s – struck text  CHANGED
samp – (sample) output
script – embedded script
section – section  NEW
select – option-selection form control
small – small print  CHANGED
source – media source  NEW
span – generic span
strong – strong importance
style – style (presentation) information
sub – subscript
summary – summary, caption, or legend for a details control  NEW
sup – superscript
table – table
tbody – table row group
td – table cell
textarea – text input area
tfoot – table footer row group
th – table header cell
thead – table heading group
time – date and/or time  NEW
title – document title
tr – table row
track – supplementary media track  NEW
ul – unordered list
var – variable or placeholder text
video – video  NEW
wbr – line-break opportunity  NEW

7. Common content models

7.1. Flow elements
7.2. Metadata elements
7.3. Phrasing elements

8. Global attributes

8.1. Core attributes
8.2. Event-handler attributes
8.3. XML attributes
1. Introduction

This document is a non-normative reference that provides details to help producers of HTML content create conformant documents, and to help others check the conformance of existing documents. It has the following design goals:

- to describe the syntax and structure of the HTML language
- to describe the semantics of HTML elements and their attributes (that is, to describe what the elements and attributes represent)
- to be clear and unambiguous
- to be as concise and readable as possible

The first two of the above design goals make this reference similar in scope to the [HTML5 for Web Authors] subset of the normative [HTML5] full specification.

Certain purposes are intentionally out of scope for this reference document; specifically, it:

- does not provide any descriptions of implementation conformance criteria for HTML consumers; in particular, it does not attempt to explain how Web browsers and other user agents are meant to process documents
- does not provide detailed information on APIs related to processing of HTML content by HTML consumers.

For implementation conformance criteria for HTML consumers and detailed information on related APIs, instead see the [HTML5] full specification.

Finally, because this document is intended for use strictly as a reference, it does not attempt to also be a tutorial or “how to” authoring guide.

2. Terminology

The following is a list of terms that are used in other sections of this reference.

**browsing context**
An environment in which UAs present documents to users. A tab or window in a Web browser typically contains a browsing context, as does an iframe, or frame elements in a frameset.

**case-insensitive match**
An exact comparison of two strings, codepoint for codepoint, except that the characters in the range A–Z and the corresponding characters in the range a–z are considered to also match.

**DOM tree**
A representation of a document based on the W3C Document Object Model (DOM), as defined in Document Object Model (DOM) Level 3 Core
specification, and as explained in the “What is the Document Object Model?” introduction for that specification [DOM Core]. UAs build representations of documents by parsing them and constructing DOM trees from them.

**fallback content**
Content associated with embedded content, to be used when the embedded content cannot itself be used (for example, when the embedded content is of an unsupported format).

**flow content**
Flow content consists of flow elements intermixed with normal character data.

**HTML producers**
HTML authors (that is, people) and applications (such as editors and content management systems) that produce HTML content.

**HTML consumers**
HTML user agents; that is, applications (such as Web browsers) which consume HTML content.

**must**
The word *must* is used in this reference to mark document-conformance requirements that are normatively defined in the HTML5: A vocabulary and associated APIs for HTML and XHTML full specification. [HTML5].

**Note:** This reference does not itself normatively define any requirements.

**newline**
Any of the following:
- a U+000D CARRIAGE RETURN (CR) character
- a U+000A LINE FEED (LF) character
- a U+000D CARRIAGE RETURN (CR) followed by a U+000A LINE FEED (LF) character

**phrasing content**
Consists of phrasing elements intermixed with normal character data.

**resolve a URL**
To resolve a URL to an absolute URL, as defined in [HTML5].

**space**
Any of the following:
- U+0020 SPACE
- U+0009 CHARACTER TABULATION (tab)
- U+000A LINE FEED (LF)
- U+000C FORM FEED (FF)
- U+000D CARRIAGE RETURN (CR)

**top-level browsing context**
As defined in [HTML5].

**transparent**
If the content model of a particular element *foo* is described as *transparent*, it means:
when element foo is a child of a parent element bar whose content model is allowed to contain flow elements, then the contents of element foo may also contain flow elements
when element foo is a child of a parent element baz whose content model restricts its child elements to only being phrasing elements, then any child elements of element foo are also restricted to only being phrasing elements

UA
An HTML user agent; that is, an application (such as a Web browser) which consumes HTML content.

3. Documents

This section defines the term document, and provides additional details related to the definition of that term. It is divided into the following parts:

1. The HTML language and HTML and XML syntaxes
2. The HTML namespace and MIME types
3. Conformant documents
4. Case insensitivity in tag names and attribute names

3.1. The HTML language and HTML and XML syntaxes

The term document is used in this reference to mean an instance of the HTML language.

The HTML language is the language described in this reference; it is an abstract language that applications can potentially represent in memory in any number of possible ways, and that can be transmitted using any number of possible concrete syntaxes.

This reference describes two particular concrete syntaxes for the HTML language: One syntax, which is referred to throughout this reference as the HTML syntax, and another syntax, which is referred to throughout this reference as the XML syntax. Web browsers typically implement two separate parsers for processing documents: an HTML parser which is invoked when processing documents in the HTML syntax, and an XML parser which is invoked when processing documents in the XML syntax.

The HTML syntax is the syntax described in the HTML syntax section of this reference.

The XML syntax is defined by rules in the [XML] specification and in the [Namespaces In XML] specification; any syntax-level requirements for documents in the XML syntax described in this reference are intended to be the same as those defined in the XML specification.

3.2. The HTML namespace and MIME types

The HTML namespace is defined as http://www.w3.org/1999/xhtml. The HTML namespace is the namespace both for documents in the HTML syntax and for documents in the XML syntax.

Documents that are served with the text/html MIME type must match the descriptions in this reference for characteristics of documents in the HTML syntax.
Documents that have an HTML namespace declaration and that are served with an XML MIME type such as text/xml, application/xml, or application/xhtml+xml must match the descriptions in this reference for characteristics of documents in the XML syntax.

3.3. Conformant documents

There are two types of conformant documents:

- conformant documents in the HTML syntax
- conformant documents in the XML syntax

3.3.1. Conformant documents in the HTML syntax

A conformant document in the HTML syntax must consist of the following parts, in the following order:

1. Optionally, a single U+FEFF BYTE ORDER MARK (BOM) character.
2. Any number of comments and space characters.
3. A doctype.
4. Any number of comments and space characters.
5. An html element, with its attributes (if any) and its contents (if any).

Note: The start tag and end tag of the html element can be omitted—as well as, possibly, the start tags and end tags of certain descendants of the html element—in which case the start tag and end tag are considered to be implied.

6. Any number of comments and space characters.

Documents in the HTML syntax must match the syntax described in the HTML syntax section of this reference.

3.3.1.1. Implied start tags and end tags

In documents in the HTML syntax, the start tags and end tags of the html element and particular descendants of the html element can be omitted. In cases where tag omission of those particular elements occurs, the document can still be considered, conceptually, to contain the elements—but with their start tags and end tags implied.

The following is an example of a document with implied start tags and end tags for the html, head, and body elements. Note that it is nevertheless a complete, valid document.

```html
<!DOCTYPE html>
<title>A relatively minimal HTML document</title>
<p>Hello World!</p>
```

The DOM tree constructed from that example by a conformant UA would look like this:

```
<DOCTYPE html>
<title>A relatively minimal HTML document</title>
<p>Hello World!</p>
```
3.3.2. Conformant documents in the XML syntax

A conformant document in the XML syntax must be a namespace-well-formed XML document, as defined in the [Namespaces in XML] specification, and its root element must be an html element.

Note: Documents in the XML syntax must not make use of any features of the HTML syntax that do not follow XML well-formedness constraints (for example, documents in the XML syntax must not use unquoted attribute value syntax and must not omit tags).

3.4. Case insensitivity in tag names and attribute names

In documents in the HTML syntax:

- Tag names for HTML elements may be written with any mix of lowercase and uppercase letters that are a case-insensitive match for the names of the elements given in the HTML elements section of this document; that is, tag names are case-insensitive.
- Attribute names for HTML elements may be written with any mix of lowercase and uppercase letters that are a case-insensitive match for the names of the attributes given in the HTML elements section of this document; that is, attribute names are case-insensitive.

In documents in the XML syntax:

- Tag names for HTML elements must exactly match the names of the elements given in the HTML elements section of this document; that is, tag names are case-sensitive.
- Attribute names for HTML elements must exactly match the names of the attributes given in the HTML elements section of this document; that is, attribute names are case-sensitive.

4. HTML syntax

This section describes the HTML syntax in detail. In places, it also notes differences between the HTML syntax and the XML syntax, but it does not describe the XML syntax in detail (the XML syntax is instead defined by rules in the [XML] specification and in the [Namespaces in XML] specification).

This section is divided into the following parts:

1. The doctype
2. Character encoding declaration
3. Elements
4. Attributes
5. Text and character data
6. Character references
7. Comments
8. SVG and MathML elements in HTML documents
9. CDATA sections in SVG and MathML contents

4.1. The doctype

A doctype (sometimes capitalized as “DOCTYPE”) is an special instruction which, for legacy reasons that have to do with processing modes in browsers, is a required part of any document in the HTML syntax; it must match the characteristics of one of the following three formats:

- normal doctype
- deprecated doctype
- legacy-tool-compatible doctype

A normal doctype consists of the following parts, in exactly the following order:

1. Any case-insensitive match for the string "<!DOCTYPE".
2. One or more space characters.
3. Any case-insensitive match for the string "HTML".
4. Optionally, one or more space characters.
5. A ">" character.

The following is an example of a conformant normal doctype.

```html
<!DOCTYPE html>
```

A deprecated doctype consists of the following parts, in exactly the following order:

1. Any case-insensitive match for the string "<!DOCTYPE".
2. One or more space characters.
3. Any case-insensitive match for the string "HTML".
4. One or more space characters.
5. Any case-insensitive match for the string "PUBLIC".
6. One or more space characters.
7. A quote mark (public ID), consisting of either a "" character or a "'" character.
8. A permitted public ID
9. A matching quote mark (public ID), identical to the quote mark (public ID) used earlier (either a "" character or a "'" character).
10. Conditionally, depending on whether it is part of a permitted-public-ID-system-ID-combination, the following parts, in exactly the following order:
    1. One or more space characters.
    2. A quote mark (system ID), consisting of either a "" character or a "'" character.
    3. A permitted system ID
    4. A matching quote mark (system ID), identical to the quote mark (system ID) used earlier (either a "" character or a "'" character).
11. Optionally, one or more space characters.
12. A ">" character.
A **permitted-public-ID-system-ID-combination** is any combination of a **public ID** (the first quoted string in the doctype) and **system ID** (the second quoted string, if any, in the doctype) such that the combination corresponds to one of the six **deprecated doctypes** in the following list of **deprecated doctypes**:

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0//EN">
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0//EN" 
"http://www.w3.org/TR/REC-html40/strict.dtd">
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" 
"http://www.w3.org/TR/html4/strict.dtd">
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" 
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" 
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

A **legacy-tool-compatible doctype** consists of the following parts, in exactly the following order:

1. Any **case-insensitive match** for the string "<!DOCTYPE".
2. One or more **space characters**.
3. Any **case-insensitive match** for the string "HTML".
4. One or more **space characters**.
5. Any **case-insensitive match** for the string "SYSTEM".
6. One or more **space characters**.
7. A **quote mark**, consisting of either a "" character or a ' ' character.
8. The literal string "about:legacy-compat".
9. A matching **quote mark**, identical to the **quote mark** used earlier (either a "" character or a ' ' character).
10. Optionally, one or more **space characters**.
11. A "" character.

The following is examples of a conformant **legacy-tool-compatible doctype**.

```html
<!doctype HTML system "about:legacy-compat">
```

### 4.2. Character encoding declaration

A **character encoding declaration** is a mechanism for specifying the character encoding used to store or transmit a document.

The following restrictions apply to character encoding declarations:

- The character encoding name given **must** be the name of the character encoding used to serialize the file.
- The value **must** be a valid character encoding name, and **must** be the preferred name for that encoding, as specified in the IANA [Character Sets] registry.
- The character encoding declaration **must** be serialized without the use of character **references** or character escapes of any kind.
- The element containing the character encoding declaration must be serialized completely within the first 512 bytes of the document.

If the document does not start with a U+FEFF BYTE ORDER MARK (BOM) character, and if its encoding is not explicitly given by a Content-Type HTTP header, then the character encoding used must be an ASCII-compatible character encoding, and, in addition, if that encoding isn't US-ASCII itself, then the encoding must be specified using a meta element with a charset attribute or a meta element in the encoding declaration state.

If the document contains a meta element with a charset attribute or a meta element in the encoding declaration state, then the character encoding used must be an ASCII-compatible character encoding.

An ASCII-compatible character encoding is one that is a superset of US-ASCII (specifically, ANSI_X3.4-1968) for bytes in the set 0x09, 0x0A, 0x0C, 0x0D, 0x20 - 0x22, 0x26, 0x27, 0x2C - 0x3F, 0x41 - 0x5A, and 0x61 - 0x7A.

Documents should not use UTF-32, JIS_C6226-1983, JIS_X0212-1990, HZ-GB-2312, JOHAB (Windows code page 1361), encodings based on ISO-2022, or encodings based on EBCDIC.

Documents must not use CESU-8, UTF-7, BOCU-1, or SCSU encodings.

In a document in the XML syntax, the XML declaration, as defined in the XML specification [XML] should be used to provide character-encoding information, if necessary.

### 4.3. Elements

An element’s content model defines the element’s structure: What contents (if any) the element can contain, as well as what attributes (if any) the element can have. The HTML elements section of this reference describes the content models for all of elements that are part of the HTML language. An element must not contain contents or attributes that are not part of its content model.

The contents of an element are any elements, character data, and comments that it contains. Attributes and their values are not considered to be the “contents” of an element.

The text content of an element is as defined in Document Object Model (DOM) Level 3 Core [DOM Core].

A void element is an element whose content model never allows it to have contents under any circumstances. Void elements can have attributes.

The following is a complete list of the void elements in HTML:

- area, base, br, col, command, embed, hr, img, input, keygen, link, meta, param, source, track, wbr

The following list describes syntax rules for the HTML syntax. Rules for the XML syntax are defined in the XML specification [XML].

- Tags are used to delimit the start and end of elements in markup. Elements have a start tag to indicate where they begin. Non-void elements have an end tag to indicate where they end.
- Tag names are used within element start tags and end tags to give the element’s name.
HTML elements all have names that only use characters in the range 0–9, a–z, and A–Z.

- **Start tags** consist of the following parts, in exactly the following order:
  1. A "<" character.
  2. The element’s *tag name*.
  3. Optionally, one or more attributes, each of which must be preceded by one or more space characters.
  4. Optionally, one or more space characters.
  5. Optionally, a "/" character, which may be present only if the element is a *void element*.
  6. A ">" character.

- **End tags** consist of the following parts, in exactly the following order:
  1. A "<" character.
  2. A "/>" character
  3. The element’s *tag name*.
  4. Optionally, one or more space characters.
  5. A ">" character.

- **Void elements** only have a start tag; end tags must not be specified for void elements.

- The start and end tags of certain elements can be omitted. The subsection for each element in the HTML elements section of this reference provides information about which tags (if any) can be omitted for that particular element.

- A non-void element must have an end tag, unless the subsection for that element in the HTML elements section of this reference indicates that its end tag can be omitted.

- The contents of an element must be placed between just after its start tag (which might be implied, in certain cases) and just before its end tag (which might be implied in certain cases).

4.3.1. Misnested tags

If an *element* has both a *start tag* and an *end tag*, its end tag must be contained within the *contents* of the same element in which its start tag is contained. An *end tag* that is not contained within the same *contents* as its *start tag* is said to be a *misnested tag*.

In the following example, the "/i" *end tag* is a *misnested tag*, because it is not contained within the *contents* of the b element that contains its corresponding "/i" *start tag*.

```
<b/foo</i>bar</b> baz</i>
```

4.4. Attributes

*Attributes* for an element are expressed inside the element’s start tag. Attributes have a *name* and a *value*.

There must never be two or more attributes on the same start tag whose names are a case-insensitive match for each other.

The following list describes syntax rules for attributes in documents in the HTML syntax. Syntax rules for attributes in documents in the XML syntax, are defined in the XML specification [XML].

- **Attribute names** must consist of one or more characters other than the *space characters*, U+0000 NULL, ",", ",", ",", ",", ",", ",", ",", the control characters, and any characters that are not
Attribute names are those that match the Name production defined in the XML specification and that contain no ":" characters, and whose first three characters are not a case-insensitive match for the string "xml".

Attribute values, in general, are normal character data; however, the HTML elements section of this reference describes further restrictions on the allowed values of particular attributes, and attributes must have values that conform to those restrictions.

In the HTML syntax, attributes can be specified in four different ways:

1. empty attribute syntax
2. unquoted attribute-value syntax
3. single-quoted attribute-value syntax
4. double-quoted attribute-value syntax

Empty attribute syntax

Certain attributes may be specified by providing just the attribute name, with no value.

In the following example, the disabled attribute is given with the empty attribute syntax:

```html
<input disabled>
```

Unquoted attribute-value syntax

An unquoted attribute value is specified by providing the following parts in exactly the following order:

1. an attribute name
2. zero or more space characters
3. a single "=" character
4. zero or more space characters
5. an attribute value

In addition to the general requirements given above for attribute values, an unquoted attribute value has the following restrictions:

- must not contain any literal space characters
- must not contain any ":", ",", ">", ",=", characters
- must not be the empty string

In the following example, the value attribute is given with the unquoted attribute value syntax:

```html
<input value=yes>
```

If the value of an attribute using the unquoted attribute syntax is followed by a "/" character, then there must be at least one space character after the value and before the "/" character.
Single-quoted attribute-value syntax

A **single-quoted attribute value** is specified by providing the following parts in exactly the following order:

1. an **attribute name**
2. zero or more **space characters**
3. a "=" character
4. zero or more **space characters**
5. a single "'" character
6. an **attribute value**
7. a "'" character.

In addition to the general requirements given above for attribute values, a single-quoted attribute value has the following restriction:

- **must** not contain any literal "'" characters

In the following example, the **type** attribute is given with the single-quoted attribute value syntax:

```html
<input type='checkbox'>
```

Double-quoted attribute-value syntax

A **double-quoted attribute value** is specified by providing the following parts in exactly the following order:

1. an **attribute name**
2. zero or more **space characters**
3. a single "=" character
4. zero or more **space characters**
5. a single "" character
6. an **attribute value**
7. a "" character

In addition to the general requirements given above for attribute values, a double-quoted attribute value has the following restriction:

- **must** not contain any literal "" characters

In the following example, the **title** attribute is given with the double-quoted attribute value syntax:

```html
<code title="U+003C LESS-T shri SIGN">&lt;</code>
```

4.5. Text and character data

**Text** in **element contents** (including in **comments**) and **attribute values** must consist of Unicode characters, with the following restrictions:
• must not contain U+0000 characters
• must not contain permanently undefined Unicode characters
• must not contain control characters other than space characters

**Character data** contains text, in some cases in combination with character references, along with certain additional restrictions. There are three types of character data that can occur in documents:

1. normal character data
2. replaceable character data
3. non-replaceable character data

**Normal character data**

Certain elements and strings in the values of particular attributes contain normal character data. Normal character data can contain the following:

• text
• character references

Normal character data has the following restrictions:

• must not contain any "<" characters

**Replaceable character data**

In documents in the HTML syntax, the title and textarea elements can contain replaceable character data. Replaceable character data can contain the following:

• text, optionally including "<" characters
• character references

Replaceable character data has the following restrictions:

• must not contain any ambiguous ampersands
• must not contain any occurrences of the string "<" followed by characters that are a case-insensitive match for the tag name of the element containing the replaceable character data (for example, "</title" or "</textarea"), followed by a space character, ">", or "/".

**Note:** Replaceable character data, as described in this reference, is a feature of the HTML syntax that is not available in the XML syntax. Documents in the XML syntax must not contain replaceable character data as described in this reference; instead they must conform to all syntax constraints described in the XML specification [XML].

**Non-replaceable character data**

In documents in the HTML syntax, the script, and style elements can contain non-replaceable character data. Non-replaceable character data can contain the following:
Non-replaceable character data has the following restrictions:

- **must** not contain character references
- **must** not contain any occurrences of the string "</", followed by characters that are a case-insensitive match for the tag name of the element containing the replaceable character data (for example, "</script" or "</style", followed by a space character, ">", or "/".

**Note:** Non-replaceable character data, as described in this reference, is a feature of the HTML syntax that is not available in the XML syntax. Documents in the XML syntax must not contain non-replaceable character data as described in this reference; instead they must conform to all syntax constraints defined in the XML specification [XML].

4.6. Character references

**Character references** are a form of markup for representing single individual characters. There are three types of character references:

- named character references
- decimal numeric character references
- hexadecimal numeric character reference

**Named character reference**

Named character references consist of the following parts in exactly the following order:

1. An "&" character.
2. One of the entity names listed in the “Named character references” section of the HTML5 specification [HTML5], using the same case.
3. A ";" character.

**Note:** For further information about named character references, see [XML Entities].

The following is an example of a named character reference for the character "†" (U+2020 DAGGER).

`&dagger;`

**Decimal numeric character reference**

Decimal numerical character references consist of the following parts, in exactly the following order:

1. An "&" character.
2. A "#" character.
3. One or more digits in the range 0–9, representing a base-ten integer that itself is a Unicode code point that is not U+0000, U+000D, in the range U+0080–U+009F, or in the range 0xD800–0xDFFF (surrogates).
4. A ";" character.

The following is an example of a decimal numeric character reference for the character "†" (U+2020 DAGGER).

\&#8224;

**Hexadecimal numeric character reference**

Hexadecimal numeric character references consist of the following parts, in exactly the following order.

1. An "&" character.
2. A "#" character.
3. Either an "x" character or an "X" character.
4. One or more digits in the range 0–9, a–f, and A–F, representing a base-sixteen integer that itself is a Unicode code point that is not U+0000, U+000D, in the range U+0080–U+009F, or in the range 0xD800–0xDFFF (surrogates).
5. A ";" character.

The following is an example of a hexadecimal numeric character reference for the character "†" (U+2020 DAGGER).

\&#x2020;

**Note:** Character references are not themselves text, and no part of a character reference is text.

An ambiguous ampersand is an "&" character that is followed by some text other than a space character, a "<", character, or another "&" character.

### 4.7. Comments

**Comments** consist of the following parts, in exactly the following order:

1. the **comment start delimiter** "<!--"
2. **text**
3. the **comment end delimiter** "-->

The **text** part of comments has the following restrictions:

- **must** not start with a ">" character
- **must** not start with the string "-->"
- **must** not contain the string "--"
must not end with a "-" character

The following is an example of a comment.

<!-- main content starts here -->

4.8. SVG and MathML elements in HTML documents

SVG and MathML elements are elements from the SVG and MathML namespaces. SVG and MathML elements can be used both in documents in the HTML syntax and in documents in the XML syntax. Syntax rules for SVG and MathML elements in documents in the XML syntax are defined in the XML specification [XML]. The following list describes additional syntax rules that specifically apply to SVG and MathML elements in documents in the HTML syntax.

- SVG and MathML elements whose start tags have a single "/" character before the closing "" character are said to be marked as self-closing.
- SVG and MathML elements must either have a start tag and an end tag, or a start tag that is marked as self-closing, in which case they must not have an end tag.
- SVG and MathML elements whose start tag is marked as self-closing, can’t have any contents.
- The contents of an SVG or MathML element whose start tag is not marked as self-closing are any elements, character data, comments, and CDATA sections that it contains, with the restriction that any character data it contains must be normal character data.

4.9. CDATA sections in SVG and MathML contents

CDATA sections in SVG and MathML contents in documents in the HTML syntax consist of the following parts, in exactly the following order:

1. the CDATA start delimiter "<![CDATA[
2. text, with the additional restriction that the text must not contain the string "]]>
3. the CDATA end delimiter "]]>"

CDATA sections are allowed only in the contents of elements from the SVG and MathML namespaces.

The following shows an example of a CDATA section.

<annotation encoding="text/latex">
  <![CDATA[\documentclass{article}
\begin{document}
\title{E}
\maketitle
The base of the natural logarithms, approximately 2.71828.
\end{document}]]>}
</annotation>

5. HTML elements organized by function
This section provides an index of HTML elements organized into groups according to function.

5.1. The root element

html – root element

5.2. Document metadata

head – document metadata container

title – document title

base – base URL

link – inter-document relationship metadata

meta – metadata

style – style (presentation) information

5.3. Scripting

script – embedded script

noscript – fallback content for script

5.4. Sections

body – document body

section – section

nav – group of navigational links

article – article

aside – tangential content

h1 – heading

h2 – heading

h3 – heading

h4 – heading

h5 – heading

h6 – heading

hgroup – heading group

header – header

footer – footer

address – contact information

5.5. Grouping content

p – paragraph

hr – thematic break

br – line break

pre – preformatted text

blockquote – block quotation

ol – ordered list

ul – unordered list

li – list item
5.6. Text-level semantics

- `a` – hyperlink [CHANGED]
- `em` – emphatic stress
- `strong` – strong importance
- `small` – small print [CHANGED]
- `cite` – cited title of a work [CHANGED]
- `q` – quoted text
- `dfn` – defining instance
- `abbr` – abbreviation
- `time` – date and/or time [NEW]
- `code` – code fragment
- `var` – variable or placeholder text
- `samp` – (sample) output
- `kbd` – user input
- `sub` – subscript
- `sup` – superscript
- `i` – offset text conventionally styled in italic [CHANGED]
- `b` – offset text conventionally styled in bold [CHANGED]
- `mark` – marked (highlighted) text [NEW]
- `ruby` – ruby annotation [NEW]
- `rt` – ruby text [NEW]
- `rp` – ruby parenthesis [NEW]
- `bdi` – BiDi isolate [NEW]
- `bdo` – BiDi override
- `span` – generic span

5.7. Edits

- `ins` – inserted text
- `del` – deleted text

5.8. Embedded content

- `img` – image
- `iframe` – nested browsing context (inline frame)
- `embed` – integration point for plugins [NEW]
- `object` – generic external content
- `param` – initialization parameters for plugins
- `video` – video [NEW]
audio – audio stream NEW
source – media source NEW
track – supplementary media track NEW
canvas – canvas for dynamic graphics NEW
map – image-map definition
area – image-map hyperlink

5.9. Tables

table – table
caption – table title
colgroup – table column group
col – table column
tbody – table row group
thead – table heading group
tfoot – table footer row group
tr – table row
td – table cell
th – table header cell

5.10. Forms

form – user-submittable form
fieldset – set of related form controls
legend – title or explanatory caption
label – caption for a form control
input – input control CHANGED
button – button
select – option-selection form control
dataList – predefined options for other controls NEW
optgroup – group of options
option – option
textarea – text input area
keygen – key-pair generator/input control NEW
output – result of a calculation in a form NEW
progress – progress indicator NEW
meter – scalar gauge NEW

5.11. Interactive elements

details – control for additional on-demand information NEW
summary – summary, caption, or legend for a details control NEW
command – command NEW
menu – list of commands CHANGED

6. HTML elements
The complete set of **HTML elements** is the set of elements described in the following sections.

- **a** – hyperlink  
- **abbr** – abbreviation  
- **address** – contact information  
- **area** – image-map hyperlink  
- **article** – article  
- **aside** – tangential content  
- **audio** – audio stream  
- **b** – offset text conventionally styled in bold  
- **base** – base URL  
- **bdi** – BiDi isolate  
- **bdo** – BiDi override  
- **blockquote** – block quotation  
- **body** – document body  
- **br** – line break  
- **button**  
  - **button** `type=submit` – submit button  
  - **button** `type=reset` – reset button  
  - **button** `type=button` – button with no additional semantics  
- **canvas** – canvas for dynamic graphics  
- **caption** – table title  
- **cite** – cited title of a work  
- **code** – code fragment  
- **col** – table column  
- **colgroup** – table column group  
- **command** – command  
  - **command** `type=command` – command with an associated action  
  - **command** `type=radio` – selection of one item from a list of items  
  - **command** `type=checkbox` – state or option that can be toggled  
- **datalist** – predefined options for other controls  
- **dd** – description or value  
- **del** – deleted text  
- **details** – control for additional on-demand information  
- **dfn** – defining instance  
- **div** – generic flow container  
- **dl** – description list  
- **dt** – term or name  
- **em** – emphatic stress  
- **embed** – integration point for plugins  
- **fieldset** – set of related form controls  
- **figcaption** – figure caption  
- **figure** – figure with optional caption  
- **footer** – footer  
- **form** – user-submittable form  
- **h1** – heading
h2 – heading
h3 – heading
h4 – heading
h5 – heading
h6 – heading
head – document metadata container
header – header NEW
hgroup – heading group NEW
hr – thematic break CHANGED
html – root element
i – offset text conventionally styled in italic CHANGED
iframe – nested browsing context (inline frame)
img – image
input – input control CHANGED
input type=text – text-input field
input type=password – password-input field
input type=checkbox – checkbox
input type=radio – radio button
input type=button – button
input type=submit – submit button
input type=reset – reset button
input type=file – file upload control
input type=hidden – hidden input control
input type=image – image-coordinates input control
input type=datatime – global date-and-time input control NEW
input type=datatime-local – local date-and-time input control NEW
input type=date – date input control NEW
input type=month – year-and-month input control NEW
input type=time – time input control NEW
input type=week – year-and-week input control NEW
input type=number – number input control NEW
input type=range – imprecise number-input control NEW
input type=email – e-mail address input control NEW
input type=url – URL input control NEW
input type=search – search field NEW
input type=tel – telephone-number-input field NEW
input type=color – color-well control NEW
ins – inserted text
kbd – user input
keygen – key-pair generator/input control NEW
label – caption for a form control
legend – title or explanatory caption
li – list item
link – inter-document relationship metadata
map – image-map definition
mark – marked (highlighted) text NEW
menu – list of commands  CHANGED
meta – metadata  CHANGED
meta name – name-value metadata
meta http-equiv=refresh – "refresh" pragma directive
meta http-equiv=default-style – "preferred stylesheet" pragma directive
meta http-equiv=content-language – "default language" pragma directive OBSOLETE
meta charset – document character-encoding declaration  NEW
meta http-equiv=content-type – document character-encoding declaration
meter – scalar gauge  NEW
nav – group of navigational links  NEW
noscript – fallback content for script
object – generic external content
ol – ordered list
optgroup – group of options
option – option
output – result of a calculation in a form  NEW
p – paragraph
param – initialization parameters for plugins
pre – preformatted text
progress – progress indicator  NEW
q – quoted text
rp – ruby parenthesis  NEW
rt – ruby text  NEW
ruby – ruby annotation  NEW
s – struck text  CHANGED
samp – (sample) output
script – embedded script
section – section  NEW
select – option-selection form control
small – small print  CHANGED
source – media source  NEW
span – generic span
strong – strong importance
style – style (presentation) information
sub – subscript
summary – summary, caption, or legend for a details control  NEW
sup – superscript
table – table
tbody – table row group
td – table cell
textarea – text input area
tfoot – table footer row group
th – table header cell
thead – table heading group
time – date and/or time  NEW
title – document title
The a element represents a hyperlink.

**Permitted contents**

**Transparent** *(either phrasing content or flow content)*

**Permitted attributes**

- **global attributes** & name & href & target & rel & hreflang & media & type

  - **name** = ID **OBSOLETE**
    
    Specifies that its a element is a **named hyperlink**, with the name given by the value of this attribute.

    **Warning:** *The name attribute on the a element is obsolete. Consider putting an id attribute on the nearest container instead.*

    **Value:** Any string, with the following restrictions:
    
    - must be at least one character long
    - must not contain any space characters

    **Note:** *Previous versions of HTML placed greater restrictions on the content of ID values (for example, they did not permit ID values to begin with a number).*

  - **href** = URL potentially surrounded by spaces
    
    A URL that provides the destination of the hyperlink. If the href attribute is not specified, the element represents a **placeholder hyperlink**.

  - **target** = browsing-context name or keyword **CHANGED**
    
    A name or keyword giving a browsing context for UAs to use when following the hyperlink.

    **Note:** *The target attribute on the a element was deprecated in a previous version of HTML, but is no longer deprecated, as it useful in*
Web applications, particularly in combination with the iframe element.

Value: Any string that is either of the following:

- a browsing-context name
- any case-insensitive match for one of the following literal strings: "_blank", "_self", "_parent", or "_top".

rel = set of space-separated tokens

A list of tokens that specify the relationship between the document containing the hyperlink and the destination indicated by the hyperlink.

hreflang = language tag

The language of the destination of the hyperlink.

Value: A valid language tag as defined in [BCP 47].

media = media-query list

The media for which the destination of the hyperlink was designed.

Value: A valid media query list as defined in [Media Queries].

type = MIME type

The MIME type of the destination of the hyperlink.

Value: A string that identifies a valid MIME media type as defined in [RFC 2046].

Additional constraints and admonitions

- The interactive element a must not appear as a descendant of the a element.
- The interactive element a must not appear as a descendant of the button element.
- The coords attribute on the a element is obsolete. For image maps, use the area element instead of the a element.
- The shape attribute on the a element is obsolete. For image maps, use the area element instead of the a element.
- The urn attribute on the a element is obsolete. Specify the preferred persistent identifier using the href attribute instead.
- The charset attribute on the a element is obsolete. Use an HTTP Content-Type header on the linked resource instead.
- The methods attribute on the a element is obsolete. Use the HTTP OPTIONS feature instead.
- The rev attribute on the a element is obsolete. Use the rel attribute instead, with a term having the opposite meaning.

Tag omission

An a element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain phrasing elements, any element that can contain flow elements

Changes in HTML5
Although previous versions of HTML restricted the a element to only containing **phrasing content** (essentially, what was in previous versions referred to as “inline” content), the a element is now **transparent**; that is, an instance of the a element is now allowed to also contain **flow content** (essentially, what was in previous versions referred to as “block” content)—if the parent element of that instance of the a element is an element that is allowed to contain **flow content**.

**Details**

If the **contents** of an a element are empty, the element represents an **empty hyperlink**.

**DOM interface**

```javascript
interface HTMLAnchorElement : HTMLElement {
  stringifier attribute DOMString href;
  attribute DOMString target;
  attribute DOMString rel;
  readonly attribute DOMTokenList relList;
  attribute DOMString media;
  attribute DOMString hreflang;
  attribute DOMString type;
  attribute DOMString text;

  // URL decomposition IDL attributes
  attribute DOMString protocol;
  attribute DOMString host;
  attribute DOMString hostname;
  attribute DOMString port;
  attribute DOMString pathname;
  attribute DOMString search;
  attribute DOMString hash;
};
```

**Typical default display properties**

```css
a:link, a:visited {
  color: (internal value);
  text-decoration: underline;
  cursor: auto;
}

a:link:active, a:visited:active {
  color: (internal value);
}
```

**Examples**
The following example is valid: A hyperlink intended to be useful in output for all media types except the `print` media type.

```html
<a href="./page2.html" rel=next media="not print">next</a>
```

The following example is obsolete: The `name` attribute is obsolete.

```html
<a name="section5"/>
```

The following example is valid: A placeholder hyperlink.

```html
<a>prev</a>
```

The following example is valid: A hyperlink that will cause its target to be opened in a new `browsing context`.

```html
<a href="http://help.example.com" target="_blank">Help</a>
```

The following example is invalid: The value of the `href` attribute contains a space character.

```html
<a href="http://example.com/Archive/Public Data/">public</a>
```

The following example is valid: A hyperlink that represents multiple link types.

```html
<a href="/../toc.html" rel="index up">up</a>
```

The following example is valid: A hyperlink that contains a paragraph.

```html
<a href="javascript:void(0);"
    onmouseover="return anno('Note: This paragraph is informative only.')"
    onmouseout="return anno_bye();">
    <p>The following is introductory information...</p>
</a>
```

---

**abbr – abbreviation**

The `abbr` element represents an abbreviation or acronym.

**Permitted contents**

- Phrasing content

**Permitted attributes**

- **global attributes**
  - Any attributes permitted globally.
Tag omission
An abbr element must have both a start tag and an end tag.

Permitted parent elements
Any element that can contain phrasing elements

Details
The title attribute on the abbr element represents the expansion of the abbreviation or acronym.

DOM interface
Uses HTMLElement.

**address – contact information**

The address element represents contact information.

Permitted contents
Flow content

Permitted attributes

- global attributes
  Any attributes permitted globally.

Additional constraints and admonitions
- The address element must not contain any nested address elements.

Tag omission
An address element must have both a start tag and an end tag.

Permitted parent elements
Any element that can contain flow elements

Details
If an address element applies to a body element, then it represents contact information for the document as a whole. If an address element applies to a section of a document, then it represents contact information for that section only.
DOM interface
Uses HTMLElement.

Typical default display properties

```javascript
address {
    display: block;
}
address {
    font-style: italic;
}
```

area – image-map hyperlink

The area element represents either a hyperlink with some text and a corresponding area on an image map, or a dead area on an image map.

Permitted contents
Empty (void element)

Permitted attributes

- global attributes
- alt & href
- target & rel & media & hreflang & type
- shape="rect" & coords
- shape="circle" & coords
- shape="poly" & coords
- shape="default"
**global attributes**

Any attributes permitted globally.

**alt** = normal character data

The fallback content for the image map.

**href** = URL potentially surrounded by spaces

A URL that provides the destination of the hyperlink for this area.

**target** = browsing-context name or keyword

A name or keyword giving a browsing context for UAs to use when following the hyperlink for this area.

Value: Any string that is either of the following:

- a browsing-context name
- any case-insensitive match for one of the following literal strings: 
  "_blank", "_self", "_parent", or "_top".

**rel** = set of space-separated tokens

A list of tokens that specify the relationship between the document containing the area and the destination indicated by the area.

**media** = media-query list

The media for which the destination of the hyperlink was designed.

Value: A valid media query list as defined in [Media Queries].

**hreflang** = language tag

The language of the destination of the hyperlink.

Value: A valid language tag as defined in [BCP 47].

**type** = MIME type

The MIME type of the destination of the hyperlink.

Value: A string that identifies a valid MIME media type as defined in [RFC 2046].

**shape** = "rect" & **coords** = rectangle coordinates

Specifies that the shape of the area of this hyperlink on the image map is a rectangle.

Value: A comma-separated list of four integers, in exactly the following order:

1. an integer representing the distance in CSS pixels from the left edge of the image to the left side of the rectangle
2. an integer representing the distance in CSS pixels from the top edge of the image to the top side of the rectangle
3. an integer, greater than the value of the first integer in this list, representing the distance in CSS pixels from the left edge of the image to the right side of the rectangle
4. an integer, greater than the value of the second integer in this list, representing the distance in CSS pixels from the top edge of the image to the bottom side of the rectangle

**shape = "circle" & coords = circle coordinates**

Specifies that the shape of the area of this hyperlink on the image map is a circle.

**Value:** A comma-separated list of three numbers, in exactly the following order:

1. an integer representing the distance in CSS pixels from the left edge of the image to the center of the circle
2. an integer representing the distance in CSS pixels from the top edge of the image to the center of the circle
3. a non-negative integer, representing the radius of the circle, in CSS pixels

**shape = "poly" & coords = polygon coordinates**

Specifies that the shape of the area of this hyperlink on the image map is a polygon.

**Value:** A comma-separated list of at least six integers, with the total number of integers in the list being even (that is, six or eight or ten numbers, and so on). Each pair of integers represents a coordinate, in CSS pixels, given as the distances from, respectively, the left and the top of the image; all the coordinates together represent the points of the polygon, in order.

**shape = "default"**

Specifies that the shape of the area of this hyperlink on the image map is a rectangle that exactly covers the entire image.

### Additional constraints and admonitions

- The `area` element must have an ancestor `map` element.
- The `nohref` attribute on the `area` element is obsolete. Omitting the `href` attribute is sufficient.

### Tag omission

The `area` element is a void element. An `area` element must have a start tag but must not have an end tag.

### Permitted parent elements

Any element that can contain phrasing elements

### DOM interface

```javascript
interface HTMLAreaElement : HTMLElement {
  attribute DOMString alt;
  attribute DOMString coords;
}
```
Typical default display properties

```javascript
area {
    display: none;
}
```

**article** – article **NEW**

The `article` element represents a section of content that forms an independent part of a document or site; for example, a magazine or newspaper article, or a blog entry.

**Permitted contents**

Zero or more `style` elements, followed by `flow content`

**Permitted attributes**

- **global attributes**
  
  Any attributes permitted globally.

**Additional constraints and admonitions**

- The sectioning element `article` must not appear as a descendant of the `address` element.

**Tag omission**

An `article` element must have both a start tag and an end tag.

**Permitted parent elements**
Any element that can contain flow elements

DOM interface
Uses HTMLElement.

Typical default display properties

```css
article {
  display: block; }
```

aside – tangential content NEW

The aside element represents content that is tangentially related to the content that forms the main textual flow of a document.

Permitted contents
Zero or more style elements, followed by flow content

Permitted attributes

global attributes
Any attributes permitted globally.

Additional constraints and admonitions

- The sectioning element aside must not appear as a descendant of the address element.

Tag omission
An aside element must have both a start tag and an end tag.

Permitted parent elements
Any element that can contain flow elements

Details

In printed documents, the type of tangential content that the aside element represents is sometimes formatted as a sidebar or annotation or footnote.

DOM interface
Uses HTMLElement.
Typical default display properties

```html
aside {
  display: block; }
```

**audio** – audio stream **NEW**

An **audio** element represents an audio stream.

**Permitted contents**

**Transparent**, with the following specific structure:

A `src` attribute, or one or more `source` elements followed by zero or more `track` elements, followed by **flow content** or

or

A `src` attribute, or one or more `source` elements followed by zero or more `track` elements, followed by **phrasing content**

**Permitted attributes**

**global attributes** & **autoplay** & **preload** & **controls** & **loop** & **src**

**global attributes**

Any attributes permitted globally.

**autoplay** = "autoplay" or "" (empty string) or empty

Instructs the UA to automatically begin playback of the audio stream as soon as it can do so without stopping.

**preload** = "none" or "metadata" or "auto" or "" (empty string) or empty

Represents a hint to the UA about whether optimistic downloading of the audio stream itself or its metadata is considered worthwhile.

- "none": Hints to the UA that the user is not expected to need the audio stream, or that minimizing unnecessary traffic is desirable.
- "metadata": Hints to the UA that the user is not expected to need the audio stream, but that fetching its metadata (duration and so on) is desirable.
- "auto": Hints to the UA that optimistically downloading the entire audio stream is considered desirable.

Specifying the empty string is equivalent to specifying the value "auto".

**controls** = "controls" or "" (empty string) or empty

Instructs the UA to expose a user interface for controlling playback of the
audio stream.

**loop** = "loop" or "" (empty string) or **empty**

Instructs the UA to seek back to the start of the audio stream upon reaching the end.

**src** = Non-empty URL potentially surrounded by spaces

The URL for the audio stream.

### Additional constraints and admonitions

- The interactive element **audio** with the attribute **controls** must not appear as a descendant of the **a** element.
- The interactive element **audio** with the attribute **controls** must not appear as a descendant of the **button** element.

### Tag omission

An **audio** element must have both a start tag and an end tag.

### Permitted parent elements

Any element that can contain **phrasing elements**, any element that can contain **flow elements**

### DOM interface

```javascript
[NamedConstructor=Audio(),
 NamedConstructor=Audio(in DOMString src)]
interface HTMLAudioElement : HTMLMediaElement {};
```

### b – offset text conventionally styled in bold

The **b** element represents a span of text offset from its surrounding content without conveying any extra emphasis or importance, and for which the conventional typographic presentation is bold text; for example, keywords in a document abstract, or product names in a review.

### Permitted contents

**Phrasing content**

### Permitted attributes

**global attributes**

Any attributes permitted globally.

### Tag omission

A **b** element must have both a start tag and an end tag.
Permitted parent elements
   Any element that can contain phrasing elements

Changes in HTML5
   Although previous versions of HTML defined the b element only in presentational terms, the
element has now been given the specific semantic purpose of representing text “offset from its
surrounding content without conveying any extra emphasis or importance, and for which the
conventional typographic presentation is bold text”.

DOM interface
   Uses HTMLElement.

Typical default display properties

```
   b {
      font-weight: bolder; }
```

base – base URL

The base element specifies a document-wide base URL for the purposes of
resolving relative URLs, and a document-wide default browsing context name
for the purposes of following hyperlinks.

Permitted contents
   Empty (void element)

Permitted attributes

```
global attributes & href & target or target
```

global attributes
   Any attributes permitted globally.

href = URL potentially surrounded by spaces
   The base URL for the document.

target = browsing-context name or keyword
   A browsing context name or keyword for use in following hyperlinks.

Value: Any string that is either of the following:
   - a browsing-context name
   - any case-insensitive match for one of the following literal strings:
     "_blank", "_self", "_parent", or "_top".
Tag omission

The base element is a void element. A base element must have a start tag but must not have an end tag.

Permitted parent elements

head

DOM interface

```javascript
interface HTMLBaseElement : HTMLElement {
    attribute DOMString href;
    attribute DOMString target;
};
```

bdi – BiDi isolate NEW

The bdi element represents a span of text that is isolated from its surroundings for the purposes of bidirectional text formatting [BIDI].

Permitted contents

Phrasing content

Permitted attributes

global attributes

Any attributes permitted globally.

Tag omission

A bdi element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain phrasing elements

DOM interface

Uses HTMLElement.

bdo – BiDi override

The bdo element represents an explicit text directionality formatting control for its children; it provides a means to specify a direction override of the Unicode BiDi algorithm [BIDI].
Permitted contents

Phrasing content

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions

- A `bdo` element must have a `dir` attribute.

Tag omission

A `bdo` element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain phrasing elements

DOM interface

Uses `HTMLElement`.

**blockquote** – block quotation

The `blockquote` element represents a section that is quoted from another source.

Permitted contents

Flow content

Permitted attributes

**global attributes** & `cite`

- **global attributes**

  Any attributes permitted globally.

  `cite = URL potentially surrounded by spaces`

  A URL referencing the original source for the quoted material.

Tag omission

A `blockquote` element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain flow elements
DOM interface

```
interface HTMLQuoteElement : HTMLElement {
    attribute DOMString cite;
};
```

Typical default display properties

```
blockquote {
    display: block;
    margin: 1em 40px 1em 40px; }
```

body – document body

The `body` element represents the body of a document (as opposed to the document’s metadata).

Permitted contents

- `Flow content`

Permitted attributes

- **global attributes**
- `onafterprint` & `onbeforeprint` & `onbeforeunload` & `onblur` & `onerror` & `onfocus` & `onhashchange` & `onload` & `onmessage` & `onoffline` & `ononline` & `onpopstate` & `onredo` & `onresize` & `onstorage` & `onundo` & `onunload`

**global attributes**

Any attributes permitted globally.

`onafterprint` = `string` NEW

User printed current document.

`onbeforeprint` = `string` NEW

User requested printing of current document.

`onbeforeunload` = `string` NEW

Document is about to be unloaded.

`onblur` = `string`

Document lost focus.

`onerror` = `string` NEW

Document failed to load properly.

`onfocus` = `string`
Document received focus.

`onhashchange = string`  NEW
Fragment identifier part of the document’s current address changed.

`onload = string`
Document finished loading.

`onmessage = string`  NEW
Document received a message.

`onoffline = string`  NEW
Network connections failed.

`ononline = string`  NEW
Network connections returned.

`onpopstate = string`  NEW
User navigated session history.

`onredo = string`  NEW
User went forward in undo transaction history.

`onresize = string`  NEW
Document view was resized.

`onstorage = string`  NEW
Storage area changed.

`onundo = string`  NEW
User went backward in undo transaction history.

`onunload = string`
Document is going away.

**Additional constraints and admonitions**

- The `alink` attribute on the `body` element is obsolete. Use CSS instead.
- The `background` attribute on the `body` element is obsolete. Use CSS instead.
- The `bgcolor` attribute on the `body` element is obsolete. Use CSS instead.
- The `link` attribute on the `body` element is obsolete. Use CSS instead.
- The `marginbottom` attribute on the `body` element is obsolete. Use CSS instead.
- The `marginheight` attribute on the `body` element is obsolete. Use CSS instead.
- The `marginleft` attribute on the `body` element is obsolete. Use CSS instead.
- The `marginright` attribute on the `body` element is obsolete. Use CSS instead.
- The `marginwidth` attribute on the `body` element is obsolete. Use CSS instead.
- The `text` attribute on the `body` element is obsolete. Use CSS instead.
- The `vlink` attribute on the `body` element is obsolete. Use CSS instead.
Tag omission

A body element’s start tag may be omitted if the first thing inside the body element is not a space character or a comment, except if the first thing inside the body element is a script or style element.

A body element’s end tag may be omitted if the body element is not immediately followed by a comment and the element is either not empty or its start tag has not been omitted.

Permitted parent elements

html

DOM interface

```
interface HTMLBodyElement : HTMLElement {
    attribute Function onafterprint;
    attribute Function onbeforeprint;
    attribute Function onbeforeunload;
    attribute Function onblur;
    attribute Function onerror;
    attribute Function onfocus;
    attribute Function onhashchange;
    attribute Function onload;
    attribute Function onmessage;
    attribute Function onoffline;
    attribute Function ononline;
    attribute Function onpopstate;
    attribute Function onpagehide;
    attribute Function onpageshow;
    attribute Function onredo;
    attribute Function onresize;
    attribute Function onscroll;
    attribute Function onstorage;
    attribute Function onundo;
    attribute Function onunload;
}
```

Typical default display properties

```
body {
    display: block;
    margin: 8px;
}

body:focus {
    outline: none;
}
```

br – line break
The `br` element represents a line break.

**Permitted contents**

Empty (void element)

**Permitted attributes**

- **global attributes**
  - Any attributes permitted globally.

**Additional constraints and admonitions**

- The `clear` attribute on the `br` element is obsolete. Use CSS instead.

**Tag omission**

The `br` element is a void element. A `br` element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain phrasing elements

**DOM interface**

```javascript
interface HTMLBRElement : HTMLElement {
}
```

### button – button

The `button` element is a multipurpose element for representing buttons.

The details of the `button` element are described in the following sections:

- `button type=submit`
- `button type=reset`
- `button type=button`

A `button` element with no `type` attribute specified represents the same thing as a `button` element with its `type` attribute set to "submit".

### button `type=submit` – submit button

The `button` element with a `type` attribute whose value is "submit" represents a button for submitting a form.

**Permitted contents**
Phrasing content

Permitted attributes

<table>
<thead>
<tr>
<th>global attributes &amp; name &amp; disabled &amp; form &amp; type &amp; value</th>
</tr>
</thead>
<tbody>
<tr>
<td>formaction &amp; autofocus &amp; formenctype &amp; formmethod &amp; formtarget &amp; formnovalidate</td>
</tr>
</tbody>
</table>

**global attributes**

Any attributes permitted globally.

**name** = string

The name part of the name/value pair associated with this element for the purposes of form submission.

**disabled** = "disabled" or "" (empty string) or empty

Specifies that the element represents a disabled control.

**form** = ID reference NEW

The value of the id attribute on the form with which to associate the element.

**type** = "submit"

Specifies that its button element is a button for submitting a form.

**value** = string

The value part of the name/value pair associated with this button for the purposes of form submission.

**formaction** = URL potentially surrounded by spaces NEW

The form-submission action for the element.

**autofocus** = "autofocus" or "" (empty string) or empty NEW

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**formenctype** = "application/x-www-form-urlencoded" or "multipart/form-data" or "text/plain" NEW

A MIME type with which a UA is meant to associate this element for form submission.

**formmethod** = "get" or "post" NEW

The HTTP method with which a UA is meant to associate this element for form submission.
formtarget = browsing-context name or keyword  \text{NEW}

A \text{browsing context} name or keyword that represents the target of the control.

Value: Any \text{string} that is either of the following:

- a \text{browsing-context name}
- any case-insensitive match for one of the following literal strings: "_blank", "_self", "_parent", or "_top".

formnovalidate = "formnovalidate" or "" (empty string) or empty  \text{NEW}

Specifies that the element represents a control whose value is not meant to be validated during form submission.

**Additional constraints and admonitions**

- The interactive element \text{button} must not appear as a descendant of the \text{a} element.
- The interactive element \text{button} must not appear as a descendant of the \text{button} element.
- Any \text{button} element descendant of a \text{label} element with a \text{for} attribute must have an ID value that matches that \text{for} attribute.

**Tag omission**

A \text{button} element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain \text{phrasing elements}

**DOM interface**

interface \text{HTMLButtonElement} : \text{HTMLElement} {
    attribute boolean \text{autofocus};
    attribute boolean \text{disabled};
    readonly attribute \text{HTMLFormElement} \text{form};
    attribute \text{DOMString} \text{formAction};
    attribute \text{DOMString} \text{formEnctype};
    attribute \text{DOMString} \text{formMethod};
    attribute \text{DOMString} \text{formNoValidate};
    attribute \text{DOMString} \text{formTarget};
    attribute \text{DOMString} \text{name};
    attribute \text{DOMString} \text{type};
    attribute \text{DOMString} \text{value};
    readonly attribute boolean \text{willValidate};
    readonly attribute \text{ValidityState} \text{validity};
    readonly attribute \text{DOMString} \text{validationMessage};
    boolean \text{checkValidity}();
    void \text{setCustomValidity} (in \text{DOMString} \text{error});
    readonly attribute \text{NodeList} \text{labels};
**button type=reset – reset button**

The button element with a type attribute whose value is "reset" represents a button for resetting a form.

**Permitted contents**

Phrasing content

**Permitted attributes**

| global attributes & name & disabled & form & type & value & autofocus |
|----------------------|---------------------------|----------------|------------------|------------------|------------------|

**global attributes**

Any attributes permitted globally.

**name = string**

The name part of the name/value pair associated with this element for the purposes of form submission.

**disabled = "disabled" or "" (empty string) or empty**

Specifies that the element represents a disabled control.

**form = ID reference**

The value of the id attribute on the form with which to associate the element.

**type = "reset"**

Specifies that its button element is a button for resetting a form.

**value = string**

The value part of the name/value pair associated with this button for the purposes of form submission.

**autofocus = "autofocus" or "" (empty string) or empty**

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**Additional constraints and admonitions**

- The interactive element button must not appear as a descendant of the a element.
- The interactive element button must not appear as a descendant of the button element.
- Any button element descendant of a label element with a for attribute must have an ID value that matches that for attribute.

**Tag omission**
A **button** element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain **phrasing elements**

**DOM interface**

```javascript
interface HTMLButtonElement : HTMLElement {
    attribute boolean autofocus;
    attribute boolean disabled;
    readonly attribute HTMLFormElement form;
    attribute DOMString formAction;
    attribute DOMString formEnctype;
    attribute DOMString formMethod;
    attribute DOMString formNoValidate;
    attribute DOMString formTarget;
    attribute DOMString name;
    attribute DOMString type;
    attribute DOMString value;
    readonly attribute boolean willValidate;
    readonly attribute ValidityState validity;
    readonly attribute NodeList labels;
}
```

**button** `type`=`button` – button with no additional semantics

The button element with a `type` attribute whose value is "button" represents a button with no additional semantics.

**Permitted contents**

**Phrasing content**

**Permitted attributes**

**global attributes**

Any attributes permitted globally.

**name** = **string**

The name part of the name/value pair associated with this element for the
purposes of form submission.

**disabled** = "disabled" or "" (empty string) or **empty**

Specifies that the element represents a disabled control.

**form** = **ID reference**  **NEW**

The value of the id attribute on the form with which to associate the element.

**type** = "button"

Specifies that its button element is a button with no additional semantics.

**value** = **string**

The value part of the name/value pair associated with this button for the purposes of form submission.

**autofocus** = "autofocus" or "" (empty string) or **empty**  **NEW**

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**Additional constraints and admonitions**

- The interactive element button must not appear as a descendant of the a element.
- The interactive element button must not appear as a descendant of the button element.
- Any button element descendant of a label element with a for attribute must have an ID value that matches that for attribute.

**Tag omission**

A button element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain phrasing elements

**DOM interface**

```java
interface HTMLButtonElement : HTMLElement {
    attribute boolean autofocus;
    attribute boolean disabled;
    readonly attribute HTMLFormElement form;
    attribute DOMString formAction;
    attribute DOMString formEnctype;
    attribute DOMString formMethod;
    attribute DOMString formNoValidate;
    attribute DOMString formTarget;
    attribute DOMString name;
    attribute DOMString type;
    attribute DOMString value;
}
```
The canvas element represents a resolution-dependent bitmap canvas, which can be used for dynamically rendering of images such as game graphics, graphs, or other images.

### Permitted contents

- **Transparent** (either phrasing content or flow content)

### Permitted attributes

- **global attributes** & **height** & **width**
  - **global attributes**
    - Any attributes permitted globally.
  - **height** = **non-negative integer**
    - The height of the canvas, in CSS pixels.
  - **width** = **non-negative integer**
    - The width of the canvas, in CSS pixels.

### Tag omission

A canvas element must have both a start tag and an end tag.

### Permitted parent elements

Any element that can contain phrasing elements, any element that can contain flow elements

### DOM interface

```java
interface HTMLCanvasElement : HTMLElement {
    attribute unsigned long width;
    attribute unsigned long height;

    DOMString toDataURL(in optional DOMString type, in any... args);
}
```
**caption – table title**

The `caption` element represents the title of the `table` that is its parent.

**Permitted contents**

- Flow content

**Permitted attributes**

- **global attributes**
  
  Any attributes permitted globally.

**Additional constraints and admonitions**

- The `align` attribute on the `caption` element is obsolete. Use CSS instead.

**Tag omission**

A `caption` element must have both a start tag and an end tag.

**Permitted parent elements**

- `table`

**DOM interface**

```javascript
interface HTMLTableCaptionElement : HTMLElement {
};
```

**Typical default display properties**

```css
caption {
    display: table-caption;
    text-align: center;
}
```

**cite – cited title of a work**

The `cite` element represents the cited title of a work; for example, the title of a book mentioned within the main text flow of a document.

**Permitted contents**

- Phrasing content
Permitted attributes

- **global attributes**
  Any attributes permitted globally.

Tag omission

A *cite* element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain *phrasing elements*.

Changes in HTML5

Although previous versions of HTML implied that the *cite* element can be used to mark up the name of a person, that usage is no longer considered conforming. The *cite* element now solely represents the cited title of a work; for example, the title of a book, paper, essay, poem, score, song, script, film, TV show, game, sculpture, painting, theater production, play, opera, musical, exhibition, legal case report, or other such work.

DOM interface

Uses [HTMLElement](http://www.w3.org/TR/2011/WD-html-markup-20110113/spec.html).

Typical default display properties

```css
  cite {
    font-style: italic;
  }
```

**code** – code fragment

The *code* element represents a fragment of computer code.

Permitted contents

- [Phrasing content](#)

Permitted attributes

- **global attributes**
  Any attributes permitted globally.

Tag omission

A *code* element must have both a start tag and an end tag.
Permitted parent elements
Any element that can contain phrasing elements

DOM interface
Uses HTMLElement.

Typical default display properties

```css
code {
  font-family: monospace;
}
```

**col** – table column

The col element represents one or more columns in the column group represented by its colgroup parent.

Permitted contents
Empty (void element)

Permitted attributes

- global attributes & span

  Any attributes permitted globally.

  **span** = positive integer

  Specifies the number of columns “spanned” by its col element; the col element shares its attributes with all the columns it spans.

Additional constraints and admonitions

- The **align** attribute on the col element is obsolete. Use CSS instead.
- The **width** attribute on the col element is obsolete. Use CSS instead.
- The **char** attribute on the col element is obsolete. Use CSS instead.
- The **charoff** attribute on the col element is obsolete. Use CSS instead.
- The **valign** attribute on the col element is obsolete. Use CSS instead.

Tag omission

The col element is a void element. A col element must have a start tag but must not have an end tag.

Permitted parent elements

colgroup
DOM interface

interface HTMLTableColElement : HTMLElement {
    attribute unsigned long span;
};

Typical default display properties

col {
    display: table-column;
}

colgroup — table column group

The `colgroup` element represents a group of one or more columns in the `table` that is its parent.

Permitted contents

Zero or more `col` elements, or an optional `span` attribute

Permitted attributes

`global attributes` & `span`

`global attributes`

Any attributes permitted globally.

`span` = `positive integer`

The number of columns in the group.

Additional constraints and admonitions

- The `width` attribute on the `colgroup` element is obsolete. [Use CSS instead.]
- The `char` attribute on the `colgroup` element is obsolete. [Use CSS instead.]
- The `charoff` attribute on the `colgroup` element is obsolete. [Use CSS instead.]
- The `valign` attribute on the `colgroup` element is obsolete. [Use CSS instead.]

Tag omission

A `colgroup` element's start tag may be omitted if the first thing inside the `colgroup` element is a `col` element, and if the element is not immediately preceded by another `colgroup` element whose end tag has been omitted.

A `colgroup` element's end tag may be omitted if the `colgroup` element is not immediately followed by a `space character` or a `comment`.

Permitted parent elements

`table`
DOM interface

```javascript
interface HTMLTableColElement : HTMLElement {
    attribute unsigned long span;
};
```

Typical default display properties

```css
colgroup {
    display: table-column-group;
}
```

**command** – command [NEW]

The command element is a multipurpose element for representing commands.

The details of the command element are described in the following sections:

*command type=command*
*command type=radio*
*command type=checkbox*

A command element with no type attribute specified represents the same thing as a command element with its type attribute set to "command".

**command type=command** – command with an associated action [NEW]

The command element with a type attribute whose value is "command" represents a command with an associated action.

Permitted contents

Empty (void element)

Permitted attributes

- global attributes & type & label & icon & disabled

  **global attributes**
  
  Any attributes permitted globally.

  **type = "command"**
  
  Specifies that its command element is a command with an associated action.

  **label = string**
The name of the command, as shown to the user.

icon = URL potentially surrounded by spaces

A URL for an image that represents the command.

disabled = "disabled" or "" (empty string) or empty

Specifies that the command is not currently available.

Tag omission

The command element is a void element. A command element must have a start tag but must not have an end tag.

Permitted parent elements

Any element that can contain metadata elements, any element that can contain phrasing elements.

DOM interface

```javascript
interface HTMLCommandElement : HTMLElement {
    attribute DOMString type;
    attribute DOMString label;
    attribute DOMString icon;
    attribute boolean disabled;
    attribute boolean checked;
    attribute DOMString radiogroup;
};
```

command type=radio – selection of one item from a list of items

The command element with a type attribute whose value is "radio" represents a selection of one item from a list of items.

Permitted contents

Empty (void element)

Permitted attributes

`global attributes` & `type` & `radiogroup` & `checked` & `label` & `icon` & `disabled`

`global attributes`

Any attributes permitted globally.

type = "radio"
Specifies that its command element represents a selection of one item from a list of items.

\textbf{radiogroup} = \texttt{string}

The name of the group of commands that will be toggled when the command itself is toggled.

\textbf{checked} = "checked" or "" (empty string) or \texttt{empty}

Specifies that the command is selected.

\textbf{label} = \texttt{string}

The name of the command, as shown to the user.

\textbf{icon} = \texttt{URL potentially surrounded by spaces}

A URL for an image that represents the command.

\textbf{disabled} = "disabled" or "" (empty string) or \texttt{empty}

Specifies that the command is not currently available.

**Tag omission**

The \texttt{command} element is a \texttt{void element}. A \texttt{command} element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain \texttt{metadata elements}, any element that can contain \texttt{phrasing elements}.

**DOM interface**

\begin{verbatim}
interface HTMLCommandElement : HTMLElement {
  attribute DOMString type;
  attribute DOMString label;
  attribute DOMString icon;
  attribute boolean disabled;
  attribute boolean checked;
  attribute DOMString radiogroup;
};
\end{verbatim}

**NEW**

\textbf{command type} = \texttt{checkbox} – state or option that can be toggled

The \texttt{command} element with a \texttt{type} attribute whose value is "\texttt{checkbox}"
represents a state or option that can be toggled.

**Permitted contents**
Empty (**void element**)

**Permitted attributes**

**global attributes** & **type** & **checked** & **label** & **icon** & **disabled**

**global attributes**

Any attributes permitted globally.

**type** = "checkbox"

Specifies that its command element represents a state or option that can be toggled.

**checked** = "checked" or "" (empty string) or **empty**

Specifies that the command is selected.

**label** = **string**

The name of the command, as shown to the user.

**icon** = Non-empty URL potentially surrounded by spaces

A URL for an image that represents the command.

**disabled** = "disabled" or "" (empty string) or **empty**

Specifies that the command is not currently available.

**Tag omission**

The command element is a **void element**. A command element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain **metadata elements**, any element that can contain **phrasing elements**

**DOM interface**

```javascript
interface HTMLCommandElement : HTMLElement {
    attribute DOMString type;
    attribute DOMString label;
    attribute DOMString icon;
    attribute boolean disabled;
    attribute boolean checked;
    attribute DOMString radiogroup;
}
```

**datalist** – predefined options for other controls **NEW**
The `datalist` element represents a set of `option` elements that represent predefined options for other controls.

**Permitted contents**

Zero or more `option` elements, intermixed with `phrasing content`.

**Permitted attributes**

- `global attributes`
  
  Any attributes permitted globally.

**Tag omission**

A `datalist` element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain `phrasing elements`.

**Details**

The `contents` of the `datalist` element represent fallback content for legacy user agents, intermixed with `option` elements that represent the predefined options.

**DOM interface**

```javascript
interface HTMLDataListElement : HTMLElement {
    readonly attribute HTMLCollection options;
};
```

**Typical default display properties**

```
datalist {
    display: none;
}
```

**dd – description or value**

The `dd` element represents a `description or value`.

**Permitted contents**

- `Flow content`

**Permitted attributes**
**global attributes**

Any attributes permitted globally.

**Tag omission**

A `dd` element must have a start tag.
A `dd` element’s end tag may be omitted if the `dd` element is immediately followed by another `dd` element or a `dd` element, or if there is no more content in the parent element.

**Permitted parent elements**

`dl`

**Details**

The `dd` element represents the **description or value** part of a term-description (name-value) grouping in a **description list**.

**DOM interface**

Uses `HTMLElement`.

**Typical default display properties**

```css
dd {
    display: block;
    margin-start: 40px;
}
```

**del – deleted text**

The `del` element represents a range of text that has been deleted from a document.

**Permitted contents**

- `Transparent` (either `phrasing content` or `flow content`)

**Permitted attributes**

- `global attributes` & `cite` & `datetime`

- **global attributes**
  
  Any attributes permitted globally.

- `cite = URL potentially surrounded by spaces`
  
  The address of a document that explains why the text was deleted.
**datetime** = **date and time** or **date**

The date and time when the text was deleted.

**Value:** A valid **date-time** as defined in [RFC 3339], with these additional qualifications:

- the literal letters **T** and **Z** in the date/time syntax **must** always be uppercase
- the **date-fullyear** production is instead defined as four or more digits representing a number greater than 0

**Examples:**

```
1990-12-31T23:59:60Z
1996-12-19T16:39:57-08:00
```

**Value:** A valid **full-date** as defined in [RFC 3339], with the additional qualification that the year component is four or more digits representing a number greater than 0.

**Example:**

```
1996-12-19
```

**Tag omission**

A **del** element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain **phrasing elements**, any element that can contain **flow elements**

**DOM interface**

```
interface HTMLModElement : HTMLElement {
    attribute DOMString cite;
    attribute DOMString dateTime;
};
```

**Typical default display properties**

```
.del {
    text-decoration: line-through;
}
```

**details** – control for additional on-demand information **NEW**

The **details** element represents a control from which the user can obtain
additional information or controls on-demand.

**Permitted contents**

An optional `summary` element, followed by `flow content`.

**Permitted attributes**

`global attributes` & `open`

- `global attributes`
  
  Any attributes permitted globally.

- `open = "open"` or `""` (empty string) or `empty`
  
  Specifies that the contents of the `details` element should be shown to the user.

**Additional constraints and admonitions**

- The interactive element `details` must not appear as a descendant of the `a` element.
- The interactive element `details` must not appear as a descendant of the `button` element.

**Tag omission**

A `details` element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain `flow elements`.

**DOM interface**

```javascript
interface HTMLDetailsElement : HTMLElement {
    attribute boolean open;
};
```

**dfn – defining instance**

The `dfn` element represents the defining instance of a term.

**Permitted contents**

`Phrasing content`

**Permitted attributes**

`global attributes`

Any attributes permitted globally.
Additional constraints and admonitions

- The dfn element must not contain any nested dfn elements.

Tag omission

A dfn element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain phrasing elements.

DOM interface

Uses HTMLElement.

**div** – generic flow container

The div element is a generic container for flow content that by itself does not represent anything.

Permitted contents

Zero or more style elements, followed by flow content.

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions

- The align attribute on the div element is obsolete. Use CSS instead.

Tag omission

A div element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain flow elements.

Details

The div can be used with attributes such as class, lang, xml:lang, and title to add additional semantics and structure to flow content.

DOM interface
Typical default display properties

```css
div {
    display: block;
}
```

**dl** – description list

The **dl** element represents a description list.

**Permitted contents**

Zero or more of: one or more **dt** elements, followed by one or more **dd** elements

**Permitted attributes**

- **global attributes**
  
  Any attributes permitted globally.

**Additional constraints and admonitions**

- The **compact** attribute on the **dl** element is obsolete. Use CSS instead.

**Tag omission**

A **dl** element must have both a **start tag** and an **end tag**.

**Permitted parent elements**

Any element that can contain flow elements

**Details**

The **dl** element represents a description list, which consists of zero or more term-description (name-value) groupings; each grouping associates one or more terms/names (the contents of **dt** elements) with one or more descriptions/values (the contents of **dd** elements).

**DOM interface**

```typescript
interface HTMLDListElement : HTMLElement {};
```
Typical default display properties

```css
dl {
  display: block;
  -webkit-margin-before: 1em;
  -webkit-margin-after: 1em;
  margin-start: 0;
  -webkit-margin-end: 0;
}
```

**dt** – term or name

The `dt` element represents a [term or name](#).

**Permitted contents**

[Phrasing content](#)

**Permitted attributes**

[global attributes](#)

Any attributes permitted globally.

**Tag omission**

A `dt` element must have a start tag.

A `dt` element’s end tag may be omitted if the `dt` element is immediately followed by another `dt` element or a `dd` element.

**Permitted parent elements**

`dl`

**Details**

The `dt` element represents the term or name part of a term-description (name-value) grouping in a description list.

**DOM interface**

Uses [HTMLElement](#).

**Typical default display properties**

```css
dt {
  display: block;
}
```
---

**em – emphatic stress**

The *em* element represents a span of text with emphatic stress.

**Permitted contents**

- Phrasing content

**Permitted attributes**

- **global attributes**
  - Any attributes permitted globally.

**Tag omission**

An *em* element must have both a start tag and an end tag.

**Permitted parent elements**

- Any element that can contain phrasing elements

**DOM interface**

- Uses [HTMLElement](http://www.w3.org/TR/2011/WD-html-markup-20110113/spec.html)

**Typical default display properties**

```css
em {
  font-style: italic;
}
```

---

**embed – integration point for plugins**

The *embed* element represents an integration point for external content.

**Permitted contents**

- Empty (void element)

**Permitted attributes**

- **global attributes** & *src* & *type* & *height* & *width* & *Any other attribute that has no namespace*

  - **global attributes**
    - Any attributes permitted globally.

  - **src** = [Non-empty URL potentially surrounded by spaces](http://www.w3.org/TR/2011/WD-html-markup-20110113/spec.html)
    - The address of the content being embedded.
**type** = MIME type
The MIME type of the plugin to instantiate.

Value: A string that identifies a valid MIME media type as defined in [RFC 2046].

**height** = non-negative integer
The height of the embedded content, in CSS pixels.

**width** = non-negative integer
The width of the embedded content, in CSS pixels.

**Additional constraints and admonitions**
- The interactive element embed must not appear as a descendant of the a element.
- The interactive element embed must not appear as a descendant of the button element.
- The name attribute on the embed element is obsolete. Use the id attribute instead.
- The align attribute on the embed element is obsolete. Use CSS instead.
- The hspace attribute on the embed element is obsolete. Use CSS instead.
- The vspace attribute on the embed element is obsolete. Use CSS instead.

**Tag omission**
The embed element is a void element. An embed element must have a start tag but must not have an end tag.

**Permitted parent elements**
Any element that can contain phrasing elements

**Details**
The embed element represents an integration point for external content — typically, non-HTML content such as an application or some other type of interactive content which involves use of a third-party plugin as a handler (rather than being natively supported by the UA).

**DOM interface**

```javascript
interface HTMLEmbedElement : HTMLElement {
    attribute DOMString src;
    attribute DOMString type;
    attribute DOMString width;
    attribute DOMString height;
};
```

**fieldset** – set of related form controls
The `fieldset` element represents a set of form controls grouped under a common name.

**Permitted contents**

An optional `legend` element, followed by `flow content`

**Permitted attributes**

`global attributes` & `name` & `disabled` & `form`

- `global attributes`
  Any attributes permitted globally.

- `name = string`
  The name part of the name/value pair associated with this element for the purposes of form submission.

- `disabled = "disabled"` or `""` (empty string) or `empty`
  Specifies that the element represents a disabled control.

- `form = ID reference` NEW
  The value of the `id` attribute on the `form` with which to associate the element.

**Tag omission**

A `fieldset` element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain `flow elements`

**DOM interface**

```javascript
interface HTMLFieldSetElement : HTMLElement {
  attribute boolean disabled;
  readonly attribute HTMLFormElement form;
  attribute DOMString name;

  readonly attribute DOMString type;

  readonly attribute HTMLFormControlsCollection elements;

  readonly attribute boolean willValidate;
  readonly attribute ValidityState validity;
  readonly attribute DOMString validationMessage;
  boolean checkValidity();
  void setCustomValidity(in DOMString error);
}
```

Typical default display properties

```
fieldset {
    display: block;
    margin-start: 2px;
    -webkit-margin-end: 2px;
    -webkit-padding-before: 0.35em;
    padding-start: 0.75em;
    -webkit-padding-end: 0.75em;
    -webkit-padding-after: 0.625em;
    border: 2px groove (internal value);
}
```

**figcaption** – figure caption  NEW

The **figcaption** element represents a caption or legend for a **figure**.

**Permitted contents**

- Flow content

**Permitted attributes**

- **global attributes**

  Any attributes permitted globally.

**Tag omission**

A **figcaption** element must have both a start tag and an end tag.

**Permitted parent elements**

- **figure**

**DOM interface**

Uses **HTMLElement**.

**figure** – figure with optional caption  NEW

The **figure** element represents a unit of content, optionally with a caption, that is self-contained, that is typically referenced as a single unit from the main flow of the document, and that can be moved away from the main flow of the document without affecting the document’s meaning.

**Permitted contents**
One `figcaption` element, followed by `flow content` or `flow content` followed by an optional `figcaption` element

### Permitted attributes

**global attributes**

Any attributes permitted globally.

### Tag omission

A `figure` element must have both a start tag and an end tag.

### Permitted parent elements

Any element that can contain `flow elements`.

### DOM interface

Uses `HTMLEElement`.

### footer – footer **NEW**

The `footer` element represents the footer for the section it applies to.

#### Permitted contents

`Flow content`

#### Permitted attributes

**global attributes**

Any attributes permitted globally.

#### Additional constraints and admonitions

- The element `footer` must not appear as a descendant of the `header` element.
- The element `footer` must not appear as a descendant of the `footer` element.
- The element `footer` must not appear as a descendant of the `address` element.

### Tag omission

A `footer` element must have both a start tag and an end tag.

### Permitted parent elements

Any element that can contain `flow elements`.

### Details
A footer typically contains information about its section such as who wrote it, links to related documents, copyright data, and the like.

**DOM interface**

Uses [HTMLElement](http).

**Typical default display properties**

```css
footer {
  display: block;
}
```

**form** — user-submittable form

The `form` element represents a user-submittable form.

**Permitted contents**

[Flow content](http)

**Permitted attributes**

- `global attributes`
- `action`
- `method`
- `enctype`
- `name`
- `accept-charset`
- `target`
- `autocomplete`
- `novalidate`

**global attributes**

Any attributes permitted globally.

**action** = [URL potentially surrounded by spaces](http)

The submission action for the form.

**method** = "get" or "post"

The HTTP method with which a UA is meant to associate this element for form submission.

**enctype** = "application/x-www-form-urlencoded" or "multipart/form-data" or "text/plain"

A MIME type with which a UA is meant to associate the form contents for form submission.

**name** = [string](http)

The name part of the name/value pair associated with this element for the purposes of form submission.

**accept-charset** = [list of character-encoding names](http)

Specifies the character encodings used for form submission.
Value: An ordered set of unique space-separated tokens, each of which must be a valid character encoding name that specifies an ASCII-compatible character encoding.

novalidate = "novalidate" or "" (empty string) or empty NEW

Specifies that the element represents a form that is not meant to be validated during form submission.

target = browsing-context name or keyword

A browsing context or keyword that represents the target of the form.

Value: Any string that is either of the following:
- a browsing-context name
- any case-insensitive match for one of the following literal strings: "_blank", "_self", "_parent", or "_top".

autocomplete = "on" or "off" NEW

Specifies whether the element represents a form for which by default a UA is meant to store the values entered into its input elements by the user (so that the UA can pre-fill the form later).

Additional constraints and admonitions
- The form element must not contain any nested form elements.

Tag omission
A form element must have both a start tag and an end tag.

Permitted parent elements
Any element that can contain flow elements

DOM interface

[OverrideBuiltins]
interface HTMLFormElement : HTMLElement {
    attribute DOMString acceptCharset;
    attribute DOMString action;
    attribute DOMString autocomplete;
    attribute DOMString enctype;
    attribute DOMString encoding;
    attribute DOMString method;
    attribute DOMString name;
    attribute boolean noValidate;
    attribute DOMString target;

    readonly attribute HTMLFormControlsCollection elements;
    readonly attribute long length;
    caller getter any (in unsigned long index);
    caller getter any (in DOMString name);
Typical default display properties

```css
form {
  display: block;
  margin-top: 0em;
}
```

### h1 – heading

The h1 through h6 elements are headings for the sections with which they are associated.

**Permitted contents**

- Phrasing content

**Permitted attributes**

- **global attributes**
  
  Any attributes permitted globally.

**Additional constraints and admonitions**

- The h1 element must not appear as a descendant of the address element.
- The align attribute on the h1 element is obsolete. Use CSS instead.

**Tag omission**

- An h1 element must have both a start tag and an end tag.

**Permitted parent elements**

- Any element that can contain flow elements, hgroup

**DOM interface**

```typescript
interface HTMLHeadingElement : HTMLElement {};
```
Typical default display properties

```
h1 {
    display: block;
    font-size: 2em;
    margin: .67em 0 .67em 0;
    font-weight: bold;
}
```

**h2 – heading**

The h1 through h6 elements are headings for the sections with which they are associated.

Permitted contents

[Phrasing content](#)

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions

- The h2 element must not appear as a descendant of the `address` element.
- The `align` attribute on the h2 element is obsolete. [Use CSS instead](#).

Tag omission

An h2 element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain flow elements, hgroup

DOM interface

```
interface HTMLHeadingElement : HTMLElement {};
```

Typical default display properties

```
h2 {
    display: block;
    font-size: 1.5em;
    margin: .83em 0 .83em 0;
    font-weight: bold;
}
```
**h3 – heading**

The h1 through h6 elements are headings for the sections with which they are associated.

**Permitted contents**

Phrasing content

**Permitted attributes**

Global attributes

Any attributes permitted globally.

**Additional constraints and admonitions**

- The h3 element must not appear as a descendant of the address element.
- The align attribute on the h3 element is obsolete. Use CSS instead.

**Tag omission**

An h3 element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain flow elements, hgroup

**DOM interface**

```javascript
interface HTMLHeadingElement : HTMLElement {};
```

**Typical default display properties**

```css
h3 {
    display: block;
    font-size: 1.17em;
    margin: 1em 0 1em 0;
    font-weight: bold; }
```

---

**h4 – heading**

The h1 through h6 elements are headings for the sections with which they are associated.

**Permitted contents**
Phrasing content

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions

- The h4 element must not appear as a descendant of the address element.
- The align attribute on the h4 element is obsolete. Use CSS instead.

Tag omission

An h4 element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain flow elements, hgroup

DOM interface

```javascript
interface HTMLHeadingElement : HTMLElement {
}
```

Typical default display properties

```css
h4 {
    display: block;
    margin: 1.33em 0 1.33em 0;
    font-weight: bold;
}
```

h5 – heading

The h1 through h6 elements are headings for the sections with which they are associated.

Permitted contents

Phrasing content

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions
• The **h5** element must not appear as a descendant of the **address** element.
• The **align** attribute on the **h5** element is obsolete. Use CSS instead.

**Tag omission**
An **h5** element must have both a start tag and an end tag.

**Permitted parent elements**
Any element that can contain **flow elements**, **hgroup**

**DOM interface**

```javascript
interface HTMLHeadingElement : HTMLElement {
}
```

**Typical default display properties**

```css
h5 {
    display: block;
    font-size: .83em;
    margin: 1.67em 0 1.67em 0;
    font-weight: bold;
}
```

**h6 – heading**

The **h1** through **h6** elements are headings for the sections with which they are associated.

**Permitted contents**

**Phrasing content**

**Permitted attributes**

**global attributes**

Any attributes permitted globally.

**Additional constraints and admonitions**

• The **h6** element must not appear as a descendant of the **address** element.
• The **align** attribute on the **h6** element is obsolete. Use CSS instead.

**Tag omission**
An **h6** element must have both a start tag and an end tag.

**Permitted parent elements**
Any element that can contain flow elements, hgroup

DOM interface

```javascript
interface HTMLHeadingElement : HTMLElement {};
```

Typical default display properties

```css
h6 {
    display: block;
    font-size: .67em;
    margin: 2.33em 0 2.33em 0;
    font-weight: bold;
}
```

**head** – document metadata container

The head element collects the document’s metadata.

Permitted contents

One title element, intermixed with an optional base element, intermixed with metadata elements

Permitted attributes

**global attributes**

Any attributes permitted globally.

Additional constraints and admonitions

- The profile attribute on the head element is obsolete. To declare which meta element terms are used in the document, instead register the names as meta extensions. To trigger specific UA behaviors, use a link element instead.

Tag omission

A head element’s start tag may be omitted if the first thing inside the head element is an element. A head element’s end tag may be omitted if the head element is not immediately followed by a space character or a comment.

Permitted parent elements

html

DOM interface

```javascript
interface HTMLHeadElement : HTMLElement {};
```
Typical default display properties

```html
head {
  display: none;
}
```

**header – header** NEW

The `header` element represents the header of a section.

### Permitted contents

**Flow content**

### Permitted attributes

**global attributes**

Any attributes permitted globally.

### Additional constraints and admonitions

- The `header` element must not appear as a descendant of the `footer` element.
- The `header` element must not appear as a descendant of the `address` element.
- The `header` element must not appear as a descendant of the `header` element.

### Tag omission

A `header` element must have both a start tag and an end tag.

### Permitted parent elements

Any element that can contain **flow elements**

### Details

The header element typically contains the headings for a section (an `h1-h6` element or `hgroup` element), along with content such as introductory material or navigational aids for the section.

### DOM interface

Uses **HTML Element**

Typical default display properties

```html
header {
```
The hgroup element represents a group of headings.

Permitted contents

One or more of:

- one h1 element, or one h2 element, or one h3 element, or one h4 element, or one h5 element, or one h6 element

Permitted attributes

- global attributes
  Any attributes permitted globally.

Tag omission

A hgroup element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain flow elements

Details

The hgroup element is typically used to group a set of one or more h1-h6 elements — to group, for example, a section title and an accompanying subtitle.

DOM interface

Uses HTMLElement.

Typical default display properties

hgroup {
  display: block;
}

hr – thematic break

The hr element represents a paragraph-level thematic break.

Permitted contents
Empty (void element)

Permitted attributes

  global attributes
  Any attributes permitted globally.

Additional constraints and admonitions

  • The align attribute on the hr element is obsolete. Use CSS instead.
  • The width attribute on the hr element is obsolete. Use CSS instead.
  • The noshade attribute on the hr element is obsolete. Use CSS instead.
  • The size attribute on the hr element is obsolete. Use CSS instead.
  • The color attribute on the hr element is obsolete. Use CSS instead.

Tag omission
The hr element is a void element. An hr element must have a start tag but must not have an end tag.

Permitted parent elements
Any element that can contain flow elements

Changes in HTML5
Although previous versions of HTML defined the hr element only in presentational terms, the element has now been given the specific semantic purpose of representing a “paragraph-level thematic break”.

Details
Some examples of thematic breaks that can be marked up using the hr element include a scene change in a story, or a transition to another topic within a section of a reference book.

DOM interface

interface HTMLHRElement : HTMLElement {};

Typical default display properties

hr {
  display: block;
html – root element

The html element represents the root of a document.

Permitted contents

One head element, followed by one body element

Permitted attributes

global attributes & manifest

global attributes

Any attributes permitted globally.

manifest = Non-empty URL potentially surrounded by spaces

The address of the document’s application cache manifest (which controls caching of content for offline use).

Additional constraints and admonitions

- The version attribute on the html element is obsolete. You can safely omit it.

Tag omission

An html element’s start tag may be omitted if the first thing inside the html element is not a comment.

An html element’s end tag may be omitted if the html element is not immediately followed by a comment and the element contains a body element that is either not empty or whose start tag has not been omitted.

DOM interface

interface HTMLHtmlElement : HTMLElement {};

Typical default display properties

html {
  display: block;
}

html:focus {
  outline: none;
}
The i element represents a span of text offset from its surrounding content without conveying any extra emphasis or importance, and for which the conventional typographic presentation is italic text; for example, a taxonomic designation, a technical term, an idiomatic phrase from another language, a thought, or a ship name.

**Permitted contents**

Phrasing content

**Permitted attributes**

*global attributes*

Any attributes permitted globally.

**Tag omission**

An i element must have both a start tag and an end tag.

**Permitted parent elements**

Any element that can contain phrasing elements

**Changes in HTML5**

Although previous versions of HTML defined the i element only in presentational terms, the element has now been given the specific semantic purpose of representing text “offset from its surrounding content without conveying any extra emphasis or importance, and for which the conventional typographic presentation is italic text”.

**DOM interface**

Uses HTMLElement.

**Typical default display properties**

```html
i {
  font-style: italic;
}
```

**iframe** – nested browsing context (inline frame)

The iframe element introduces a new nested browsing context.

**Permitted contents**

Normal character data
Permitted attributes

**global attributes**

Any attributes permitted globally.

**src** = Non-empty URL potentially surrounded by spaces

The address of a page that the nested browsing context is to contain.

**name** = browsing-context name

A valid browsing-context name.

Value: Any string, with the following restrictions:

- must not start with a "_" character
- must be at least one character long

**width** = non-negative integer

The width of the iframe, in CSS pixels.

**height** = non-negative integer

The height of the iframe, in CSS pixels.

**sandbox** = sandbox “allow” keywords list

NEW

Instructs the UA to “sandbox” the iframe by disallowing/disabling various features, and then (optionally) re-allowing/re-enabling particular features.

The presence of this attribute—even if no value is given (that is, the attribute is specified using empty attribute syntax) or even if the value consists only of space characters—instructs the UA to enforce all the following default sandbox restrictions:

- **plugins** are disallowed/disabled within the iframe
- **forms** are disallowed/disabled within the iframe
- **scripts** are disallowed/disabled within the iframe
- **links to other browsing contexts** are disallowed/disabled within the iframe
- **same-origin treatment** of the content hosted by the iframe and the content which hosts the iframe is disallowed/disabled; the iframe is instead treated as being from a unique origin

If any sandbox “allow” keywords are given, they instruct the UA to override particular default sandbox restrictions by re-allowing/re-enabling certain features, as follows:

- **allow-forms**: re-enables forms
- **allow-scripts**: re-enables scripts
- **allow-top-navigation**: re-enables links within the content of the iframe to navigate to its top-level browsing context
- **allow-same-origin**: re-enables same-origin treatment of the content hosted by the iframe and the content which hosts the iframe (instead of forcing the iframe into a unique origin)
seamless = "seamless" or "" (empty string) or empty  NEW

Instructs the UA that the iframe element’s browsing context is to be rendered in a manner that makes it appear to be part of the containing document (seamlessly included in the parent document).

Additional constraints and admonitions

- The interactive element iframe must not appear as a descendant of the a element.
- The interactive element iframe must not appear as a descendant of the button element.
- The longdesc attribute on the iframe element is obsolete. Use a regular a element to link to the description.
- The align attribute on the iframe element is obsolete. Use CSS instead.
- The allowtransparency attribute on the iframe element is obsolete. Use CSS instead.
- The frameborder attribute on the iframe element is obsolete. Use CSS instead.
- The marginheight attribute on the iframe element is obsolete. Use CSS instead.
- The marginwidth attribute on the iframe element is obsolete. Use CSS instead.
- The scrolling attribute on the iframe element is obsolete. Use CSS instead.

Tag omission

An iframe element must have both a start tag and an end tag.

Permitted parent elements

Any element that can contain phrasing elements

DOM interface

interface HTMLIFrameElement : HTMLElement {
    attribute DOMString src;
    attribute DOMString srcdoc;
    attribute DOMString name;
    [PutForwards=value] readonly attribute DOMSettableTokenList sandbox;
    attribute boolean seamless;
    attribute DOMString width;
    attribute DOMString height;
    readonly attribute Document contentDocument;
    readonly attribute WindowProxy contentWindow;
};

Typical default display properties

iframe {
    border: 2px inset;
}

img – image
The **img** element represents an image.

**Permitted contents**

Empty ([void element](http://www.w3.org/TR/2011/WD-html-markup-20110113/spec.html#void-element))

**Permitted attributes**

`global attributes` & `src` & `alt` & `height` & `width` & `usemap` & `ismap` & `border`

**global attributes**

Any attributes permitted globally.

**src** = Non-empty URL potentially surrounded by spaces

A URL referencing a non-interactive, optionally animated, image that is neither paged nor scripted.

**alt** = normal character data

The fallback content for the image.

**height** = non-negative integer

The height of the image, in CSS pixels.

**width** = non-negative integer

The width of the image, in CSS pixels.

**usemap** = hash-name reference

A hash-name reference to a map element with which to associate the image.

**ismap** = "ismap" or "" (empty string) or empty

Specifies that its `img` element provides access to a server-side image map.

**border** = zero  OBSOLETE

Specifies that its `img` element represents a borderless image.

**Warning:** The `border` attribute on the `img` element is obsolete. Consider specifying "img { border: 0; }" in CSS instead.

Value: The literal string ",0".

**Additional constraints and admonitions**

- The element `img` with the attribute `usemap` must not appear as a descendant of the `a` element.
- The element `img` with the attribute `usemap` must not appear as a descendant of the `button` element.
- The `img` element with the `ismap` attribute set must have an ancestor `a` element with the `href` attribute.
- The `longdesc` attribute on the `img` element is obsolete. Use a regular `a` element to link to the
Tag omission

The **img** element is a **void element**. An **img** element must have a start tag but must not have an end tag.

Permitted parent elements

Any element that can contain **phrasing elements**

### DOM interface

```javascript
[NamedConstructor=Image(),
 NamedConstructor=Image(in unsigned long width),
 NamedConstructor=Image(in unsigned long width, in unsigned long height)]
interface HTMLImageElement : HTMLElement {
    attribute DOMString alt;
    attribute DOMString src;
    attribute DOMString useMap;
    attribute boolean isMap;
    attribute unsigned long width;
    attribute unsigned long height;
    readonly attribute unsigned long naturalWidth;
    readonly attribute unsigned long naturalHeight;
    readonly attribute boolean complete;
}
```

### input – input control

The **input** element is a multipurpose element for representing input controls.

The details of the **input** element are described in the following sections:

- **input type=text**
- **input type=password**
- **input type=checkbox**
- **input type=radio**
- **input type=button**
- **input type=submit**
- **input type=reset**
- **input type=file**
- **input type=hidden**
An `<input>` element with no `type` attribute specified represents the same thing as an `<input>` element with its `type` attribute set to "text".

Changes in HTML5

Several new `<input>` element types have been added, and several new attributes are now allowed on the element.

```html
<input type=text> – text-input field
```

The `<input>` element with a `type` attribute whose value is "text" represents a one-line plain text edit control for the `<input>` element's `value`.

Permitted contents

Empty (void element)

Permitted attributes

```
| global attributes & name & disabled & form & type & maxlength & readonly & |
| size & value |

&

| autocomplete & autofocus & list & pattern & required & placeholder |
```

**global attributes**

Any attributes permitted globally.

**name** = `string`

The name part of the name/value pair associated with this element for the
purposes of form submission.

**disabled** = "disabled" or "" (empty string) or empty

Specifies that the element represents a disabled control.

**form** = **ID reference**

The value of the id attribute on the form with which to associate the element.

**type** = "text"

Specifies that its input element is a one-line plain-text edit control for the input element’s value.

**maxlength** = **positive integer**

The maximum allowed value length of the element.

**readonly** = "readonly" or "" (empty string) or empty

Specifies that element represents a control whose value is not meant to be edited.

**size** = **positive integer**

The number of options meant to be shown by the control represented by its element.

**value** = **string**

Specifies a value for this input element.

**autocomplete** = "on" or "off"  **NEW**

Specifies whether the element represents an input control for which a UA is meant to store the value entered by the user (so that the UA can prefill the form later).

**autofocus** = "autofocus" or "" (empty string) or empty  **NEW**

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**list** = **ID reference**  **NEW**

The value of the id attribute on the datalist with which to associate the element.

**pattern** = **pattern**  **NEW**

Specifies a regular expression against which a UA is meant to check the value of the control represented by its element.

**Value:** A regular expression that must match the JavaScript Pattern production as specified in [ECMA 262].

**required** = "required" or "" (empty string) or empty  **NEW**

Specifies that the element is a required part of form submission.
placeholder = string  \[NEW\]

A short hint (one word or a short phrase) intended to aid the user when entering data into the control represented by its element.

**Additional constraints and admonitions**

- The interactive element `input` must not appear as a descendant of the `a` element.
- The interactive element `input` must not appear as a descendant of the `button` element.
- Any `input` element descendant of a `label` element with a `for` attribute must have an ID value that matches that `for` attribute.
- The `list` attribute of the `input` element must refer to a `datalist` element.
- Element `input` with attribute `type` whose value is “button” must have non-empty attribute `value`.
- The `usemap` attribute on the `input` element is obsolete. Use the `img` element instead of the `input` element for image maps.
- The `align` attribute on the `input` element is obsolete. Use CSS instead.

**Tag omission**

The `input` element is a **void element**. An `input` element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain phrasing elements

**DOM interface**

```javascript
interface HTMLInputElement : HTMLElement {
  attribute DOMString accept;
  attribute DOMString alt;
  attribute DOMString autocomplete;
  attribute boolean autofocus;
  attribute boolean defaultChecked;
  attribute boolean checked;
  attribute DOMString dirName;
  attribute boolean disabled;
  readonly attribute HTMLFormElement form;
  readonly attribute FileList files;
  attribute DOMString formAction;
  attribute DOMString formEnctype;
  attribute DOMString formMethod;
  attribute boolean formNoValidate;
  attribute DOMString formTarget;
  attribute DOMString height;
  attribute boolean indeterminate;
  readonly attribute HTMLElement list;
  attribute DOMString max;
  attribute long maxLength;
  attribute DOMString min;
```
The input element with a type attribute whose value is "password" represents a one-line plain-text edit control for entering a password.

Permitted contents
Empty (void element)

Permitted attributes

```html
attribute boolean multiple;
attribute DOMString name;
attribute DOMString pattern;
attribute DOMString placeholder;
attribute boolean readOnly;
attribute boolean required;
attribute unsigned long size;
attribute DOMString src;
attribute DOMString step;
attribute DOMString type;
attribute DOMString defaultValue;
attribute DOMString value;
attribute Date valueAsDate;
attribute double valueAsNumber;
readonly attribute HTMLOptionElement selectedOption;
attribute DOMString width;

void stepUp(in optional long n);
void stepDown(in optional long n);

readonly attribute boolean willValidate;
readonly attributeValidityState validity;
readonly attribute DOMString validationMessage;
boolean checkValidity();
void setCustomValidity(in DOMString error);

readonly attribute NodeList labels;

void select();
    attribute unsigned long selectionStart;
    attribute unsigned long selectionEnd;
void setSelectionRange(in unsigned long start, in unsigned long end);
```
global attributes

Any attributes permitted globally.

name = string

The name part of the name/value pair associated with this element for the purposes of form submission.

disabled = "disabled" or "" (empty string) or empty

Specifies that the element represents a disabled control.

form = ID reference  NEW

The value of the id attribute on the form with which to associate the element.

type = "password"

Specifies that its input element is a one-line plain-text edit control for entering a password.

maxlength = positive integer

The maximum allowed value length of the element.

readonly = "readonly" or "" (empty string) or empty

Specifies that element represents a control whose value is not meant to be edited.

size = positive integer

The number of options meant to be shown by the control represented by its element.

value = string

Specifies a value for this input element.

autocomplete = "on" or "off"  NEW

Specifies whether the element represents an input control for which a UA is meant to store the value entered by the user (so that the UA can prefill the form later).

autofocus = "autofocus" or "" (empty string) or empty  NEW

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

pattern = pattern  NEW

Specifies a regular expression against which a UA is meant to check the
value of the control represented by its element.

**Value:** A regular expression that **must** match the JavaScript **Pattern** production as specified in [ECMA 262].

required = "required" or "" (empty string) or empty **NEW**

Specifies that the element is a required part of form submission.

**placeholder = string **NEW**

A short hint (one word or a short phrase) intended to aid the user when entering data into the control represented by its element.

**Additional constraints and admonitions**

- The interactive element *input* must not appear as a descendant of the *a* element.
- The interactive element *input* must not appear as a descendant of the *button* element.
- Any *input* element descendant of a *label* element with a for attribute must have an ID value that matches that for attribute.
- The list attribute of the *input* element must refer to a datalist element.
- Element *input* with attribute type whose value is "button" must have non-empty attribute value.
- The usemap attribute on the *input* element is obsolete. Use the *img* element instead of the *input* element for image maps.
- The align attribute on the *input* element is obsolete. **Use CSS instead.**

**Tag omission**

The *input* element is a **void element**. An *input* element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain **phrasing elements**

**DOM interface**

```javascript
interface HTMLInputElement : HTMLElement {
    attribute DOMString accept;
    attribute DOMString alt;
    attribute DOMString autocomplete;
    attribute boolean autofocus;
    attribute boolean defaultChecked;
    attribute boolean checked;
    attribute DOMString dirName;
    attribute boolean disabled;
    readonly attribute HTMLFormElement form;
    readonly attribute FileList files;
    attribute DOMString formAction;
    attribute DOMString formEncType;
    attribute DOMString formMethod;
    attribute boolean formNoValidate;
}
```
The input element with a type attribute whose value is "checkbox" represents a state or option that can be toggled.

### Permitted contents
Empty (**void element**)

**Permitted attributes**

<table>
<thead>
<tr>
<th>global attributes</th>
<th>name</th>
<th>disabled</th>
<th>form</th>
<th>type</th>
<th>checked</th>
<th>value</th>
<th>autofocus</th>
<th>required</th>
</tr>
</thead>
</table>

**global attributes**

Any attributes permitted globally.

**name = string**

The name part of the name/value pair associated with this element for the purposes of form submission.

**disabled = "disabled" or "" (empty string) or empty**

Specifies that the element represents a disabled control.

**form = ID reference NEW**

The value of the id attribute on the form with which to associate the element.

**type = "checkbox"**

Specifies that its input element represents a state or option that can be toggled.

**checked = "checked" or "" (empty string) or empty**

Specifies that the element represents a selected control.

**value = string**

Specifies a value for the input element.

**autofocus = "autofocus" or "" (empty string) or empty NEW**

Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**required = "required" or "" (empty string) or empty NEW**

Specifies that the element is a required part of form submission.

**Additional constraints and admonitions**

- The interactive element input must not appear as a descendant of the a element.
- The interactive element input must not appear as a descendant of the button element.
- Any input element descendant of a label element with a for attribute must have an ID value that matches that for attribute.
- The list attribute of the input element must refer to a datalist element.
- Element input with attribute type whose value is "button" must have non-empty attribute value.
- The usemap attribute on the input element is obsolete. Use the img element instead of the input element for image maps.
- The align attribute on the input element is obsolete. [Use CSS instead](http://www.w3.org/TR/2011/WD-html-markup-20110113/spec.html[16/01/2011 17:28:48])
Tag omission

The input element is a void element. An input element must have a start tag but must not have an end tag.

Permitted parent elements

Any element that can contain phrasing elements

DOM interface

```javascript
interface HTMLInputElement : HTMLElement {
  attribute DOMString accept;
  attribute DOMString alt;
  attribute DOMString autocomplete;
  attribute boolean autofocus;
  attribute boolean defaultChecked;
  attribute boolean checked;
  attribute DOMString dirName;
  attribute boolean disabled;
  readonly attribute HTMLFormElement form;
  readonly attribute FileList files;
  attribute DOMString formAction;
  attribute DOMString formEnctype;
  attribute DOMString formMethod;
  attribute boolean formNoValidate;
  attribute DOMString formTarget;
  attribute DOMString height;
  attribute boolean indeterminate;
  attribute DOMString list;
  attribute DOMString max;
  attribute long maxLength;
  attribute DOMString min;
  attribute boolean multiple;
  attribute DOMString name;
  attribute DOMString pattern;
  attribute DOMString placeholder;
  attribute boolean readOnly;
  attribute boolean required;
  attribute unsigned long size;
  attribute DOMString src;
  attribute DOMString step;
  attribute DOMString type;
  attribute DOMString defaultValue;
  attribute DOMString value;
  attribute Date valueAsDate;
  attribute double valueAsNumber;
  readonly attribute HTMLOptionElement selectedOption;
  attribute DOMString width;
```
void stepUp(in optional long n);
void stepDown(in optional long n);

readonly attribute boolean willValidate;
readonly attribute ValidityState validity;
readonly attribute DOMString validationMessage;
boolean checkValidity();
void setCustomValidity(in DOMString error);

readonly attribute NodeList labels;

void select();
    attribute unsigned long selectionStart;
    attribute unsigned long selectionEnd;
void setSelectionRange(in unsigned long start, in unsigned long end);
};

input type=radio – radio button

The input element with a type attribute whose value is "radio" represents a selection of one item from a list of items (a radio button).

Permitted contents
Empty (void element)

Permitted attributes

- global attributes & name & disabled & form & type & checked & value & autofocus & required

- global attributes
Any attributes permitted globally.

name = string
The name part of the name/value pair associated with this element for the purposes of form submission.

disabled = "disabled" or "" (empty string) or empty
Specifies that the element represents a disabled control.

form = ID reference NEW
The value of the id attribute on the form with which to associate the element.

type = "radio"
Specifies that its input element represents a selection of one item from a list of items.
**checked** = "checked" or "" (empty string) or empty
Specifies that the element represents a selected control.

**value** = string
Specifies a value for the input element.

**autofocus** = "autofocus" or "" (empty string) or empty  NEW
Specifies that the element represents a control to which a UA is meant to give focus as soon as the document is loaded.

**required** = "required" or "" (empty string) or empty  NEW
Specifies that the element is a required part of form submission.

**Additional constraints and admonitions**

- The interactive element input must not appear as a descendant of the a element.
- The interactive element input must not appear as a descendant of the button element.
- Any input element descendant of a label element with a for attribute must have an ID value that matches that for attribute.
- The list attribute of the input element must refer to a datalist element.
- Element input with attribute type whose value is "button" must have non-empty attribute value.
- The usemap attribute on the input element is obsolete. Use the img element instead of the input element for image maps.
- The align attribute on the input element is obsolete. Use CSS instead.

**Tag omission**

The input element is a void element. An input element must have a start tag but must not have an end tag.

**Permitted parent elements**

Any element that can contain phrasing elements

**DOM interface**

```javascript
interface HTMLInputElement : HTMLElement {
    attribute DOMString accept;
    attribute DOMString alt;
    attribute DOMString autocomplete;
    attribute boolean autofocus;
    attribute boolean defaultChecked;
    attribute boolean checked;
    attribute DOMString dirName;
    attribute boolean disabled;
    readonly attribute
```
