HTML Working Group
INTERNET-DRAFT
<draft-ietf-html-spec-02.ps>
Expires September 29, 1995

T. Berners-Lee
D. Connolly
MIT/W3C
March 29, 1995

HyperText Markup Language — 2.0

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

To learn the current status of any Internet-Draft, please check the "`1id-abstracts.txt`" listing contained in the Internet-Drafts Shadow Directories on `ftp.is.co.za` (Africa), `nic.nordu.net` (Europe), `munnari.oz.au` (Pacific Rim), `ds.internic.net` (US East Coast), or `ftp.isi.edu` (US West Coast).

Distribution of this document is unlimited. Please send comments to the HTML working group (HTML-WG) of the Internet Engineering Task Force (IETF) at <html-wg@oclc.org>. Discussions of the group are archived at <URL:http://www.acl.lanl.gov/HTML_WG/archives.html>.

Abstract

The HyperText Markup Language (HTML) is a simple markup language used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of applications. HTML markup can represent hypertext news, mail, documentation, and hypermedia; menus of options; database query results; simple structured documents with in-lined graphics; and hypertext views of existing bodies of information.

HTML has been in use by the World Wide Web (WWW) global information initiative since 1990. This specification roughly corresponds to the capabilities of HTML in common use prior to June 1994. It is defined as an application of ISO Standard 8879:1986 Information Processing Text and Office Systems; Standard Generalized Markup Language (SGML).

The "text/html; version=2.0" Internet Media Type (RFC 1590) and MIME Content Type (RFC 1521) is defined by this specification.

Contents

1. Introduction 5
   1.1 Purpose 5
   1.2 Levels of Conformance 5
   1.3 Terminology 5
   1.4 Imperatives 7
2. HTML as an Application of SGML 7
   2.1 SGML Documents 7
   2.2 SGML Syntax 7
      2.2.1 Elements 8
      2.2.2 Names 8
      2.2.3 Attributes 8
      2.2.4 Entity References 9
      2.2.5 Comments 9
   2.3 Example HTML Document 10
3. HTML as an Internet Media Type 11
   3.1 text/html media type 11
   3.2 Character Set Issues 11
   3.3 Undefined Tag and Attribute Names 12
4. Document Structure Elements 12
   4.1 HTML Identifier 12
   4.2 Head 13
   4.3 Body 13
5. Document Metainformation Elements 13
   5.1 Title 13
   5.2 Base 13
   5.3 Isindex 13
   5.4 Link 14
   5.5 Meta 14
   5.6 Nextid 15
6. Data Characters 16
   6.1 Special Characters 16
   6.2 Control Characters 16
   6.3 Character Entities 17
      6.3.1 Character Name References 17
      6.3.2 Character Octet References 17
7. Data Elements 18
   7.1 Line Break 18
   7.2 Horizontal Rule 18
   7.3 Image 18
8. Character Format Elements 19
   8.1 Semantic Format Elements 19
      8.1.1 Citation 19
      8.1.2 Code 20
      8.1.3 Emphasis 20

8.1.4 Keyboard ... 20
8.1.5 Sample ... 20
8.1.6 Strong ... 20
8.1.7 Variable ... 20
8.2 Physical Format Elements ... 20
8.2.1 Bold ... 20
8.2.2 Italic ... 20
8.2.3 Teletype ... 20
9. Hypertext Elements ... 21
9.1 Anchor ... 21
9.1.1 HREF ... 21
9.1.2 NAME ... 21
9.1.3 TITLE ... 22
9.1.4 REL ... 22
9.1.5 REV ... 22
9.1.6 URN ... 22
9.1.7 METHODS ... 22
10. Block Structuring Elements ... 22
10.1 Paragraph ... 22
10.2 Preformatted Text ... 23
10.3 Address ... 24
10.4 Blockquote ... 24
10.5 Headings ... 24
10.6 List Elements ... 25
10.6.1 Definition List ... 25
10.6.2 Directory List ... 26
10.6.3 Menu List ... 26
10.6.4 Ordered List ... 26
10.6.5 Unordered List ... 27
11. Form-based Input Elements ... 28
11.1 Form ... 29
11.2 Input ... 29
11.3 Option ... 31
11.4 Select ... 32
11.5 Text Area ... 32
12. HTML Document Type Definitions ... 33
12.1 Deprecated and Recommended Sections ... 33
12.2 Sample SGML Open Entity Catalog for HTML ... 33
12.3 SGML Declaration for HTML ... 34
12.4 HTML Level 0 DTD ... 35
12.5 HTML Level 1 DTD ... 36
12.6 HTML Level 2 DTD ... 36
13. Character Entity Sets ... 48
13.1 Numeric and Special Graphic Entity Set ... 48
13.2 ISO Latin 1 Character Entity Set ... 48
13.3 Character Octet Entity Set ... 50
14. Security Considerations ... 53
15. References ... 53

16. Acknowledgments 54
17. Authors' Addresses 54
Appendix A. Obsolete Features.....55
A.1 Comment Element 55
A.2 Highlighted Phrase Element 55
A.3 Plain Text Element.....55
A.4 Example and Listing Elements 55
Appendix B. Proposed Features 56
B.1 Additional Character Entities 56
B.2 Defining Instance Element.....56
B.3 Strike Element 56
B.4 Underline Element 57

1. Introduction

1.1 Purpose

The HyperText Markup Language (HTML) is a simple markup language used to create hypertext documents that are portable from one platform to another. HTML documents are SGML documents with generic semantics that are appropriate for representing information from a wide range of applications. HTML has been in use by the World-Wide Web (WWW) global information initiative since 1990. This specification corresponds to the capabilities of HTML in common use prior to June 1994 and referred to as “HTML 2.0”.

This specification defines HTML as an application of ISO Standard 8879:1986 Information Processing Text and Office Systems; Standard Generalized Markup Language (SGML). SGML provides a formal definition of the HTML syntax in the form of a Document Type Definition (DTD).

This specification also defines HTML as an Internet Media Type [7] and MIME Content Type [4] called "text/html", or "text/html; version=2.0". As such, it defines the semantics of the HTML syntax and how that syntax should be interpreted by user agents.

1.2 Levels of Conformance

Version 2.0 of HTML introduces a distinction between levels of conformance:

Level 0
: Indicates the minimum conformance level. When writing Level 0 documents, authors can be confident that the rendering at different sites will reflect their intent.

Level 1
: Includes Level 0 features plus features such as highlighting and images.

Level 2
: Includes all Level 0 and Level 1 features, plus forms.

1.3 Terminology

The HTML specification uses these words with precise meanings:

attribute
: A syntactical component of an HTML element which is often used to specify a characteristic quality of an element, other than type or content.

document type definition (DTD)
: A DTD is a collection of declarations (entity, element, attribute, link, map, etc.) in SGML syntax that defines the components and structures available for a class (type) of documents.

element
: A component of the hierarchical structure defined by the document type definition; it is identified in a document instance by descriptive markup, usually a start-tag and an end-tag.

HTML document
: A collection of information represented as a sequence of characters. An HTML document consists of data characters and markup. In particular, the markup describes a structure conforming to the HTML document type definition.

HTTP
: The Hypertext Transfer Protocol [3] is the primary application-level protocol for the transfer of documents via the World-Wide Web.

interpreter
: A tool or algorithm used to parse and/or render an HTML document.

(document) instance
: The document itself including the actual content with the actual markup. Can be a single document or part of a document instance set that follows the DTD.

markup
: Text added to the data of a document to convey information about it. There are four different kinds of markup: descriptive markup (tags), references, markup declarations, and processing instructions.

MIME
: The Multipurpose Internet Mail Extensions [4] provide the ability to transfer non-textual data, such as graphics, audio and fax, via Internet mail.

representation
: The encoding of information for interchange. For example, HTML is a representation of hypertext.

rendering
: Formatting and presenting information.

SGML
: Standard Generalized Markup Language [12] (see also [9] and [6]) is a programming language for describing the allowed markup constructs and syntax for textual document types.

tag
: Descriptive markup. There are two kinds of tags; start-tags and end-tags. All HTML tags start with the less than character ("<") and end with a greater than (">").

URI
: A Universal Resource Identifier [1] is a formatted string that serves as an identifier for a resource on the Internet. URIs are used by HTML to identify the destination of hypertext links, the source of in-line images, and the object of form actions. URIs in common use include Uniform Resource Locators (URLs) [2] and Relative URLs [5].

WWW
: The World-Wide Web is a hypertext-based, distributed information system created by researchers at CERN in Switzerland. Users may create, edit or browse hypertext documents. <URL:http://www.w3.org/>

1.4 Imperatives

may
: The implementation is not obliged to follow this in any way.

must
: If this is not followed, the implementation does not conform to this specification.

shall
: If this is not followed, the implementation does not conform to this specification.

should
: If this is not followed, though the implementation officially conforms to the specification, undesirable results may occur in practice.

typical
: Typical rendering is described for many elements. This is not a mandatory part of the specification but is given as guidance for designers and to help explain the uses for which the elements were intended.

2. HTML as an Application of SGML

HTML is an application of ISO Standard 8879:1986 — Standard Generalized Markup Language (SGML) [12]. SGML is a system for defining structured document types and markup languages to represent instances of those document types. The SGML declaration for HTML and the HTML document type definitions (DTDs) are provided in Section 12.

2.1 SGML Documents

Every SGML document has three parts:

SGML declaration
: Binds SGML processing quantities and syntax token names to specific values. For example, the SGML declaration in the HTML DTD specifies that the string that opens an end tag is "</" and the maximum length of a name is 72 characters.

Prologue
: Includes one or more document type declarations, which specify the element types, element relationships and attributes.

Instance
: Contains the data and markup of the document.

The term “HTML” refers to both the document type and the markup language for representing instances of that document type.

2.2 SGML Syntax

An HTML document instance is a text file in which some of the characters are markup. Markup (tags) define the structure of the document.

2.2.1 Elements

Tags define the start and end of headings, paragraphs, lists, character highlighting and links. Most HTML elements are identified in a document as a start tag, which gives the element name and attributes, followed by the content, followed by the end tag. Start tags are delimited by "<" and ">"; end tags are delimited by "</" and ">". An example is:

```
<H1>This is a Heading</H1>
```

Some elements only have a start tag without an end tag. For example, to create a line break, you use the <BR> tag. Additionally, the end tags of some other elements, such as Paragraph (<P>), List Item (<LI>), Definition Term (<DT>), and Definition Description (<DD>) elements, may be omitted.

The content of an element is a sequence of characters and nested elements. Some elements, such as anchors, cannot be nested. Anchors and character highlighting may be put inside other constructs.

> ***Note:*** *The SGML declaration for HTML specifies SHORTTAG YES, which means that there are other valid syntaxes for tags, such as NET tags, "<EM/.../"; empty start tags, "<>"; and empty end tags, "</>". Until support for these idioms is widely deployed, their use is strongly discouraged.*

2.2.2 Names

A name consists of a letter followed by up to 71 letters, digits, periods, or hyphens. Element names are not case sensitive, but entity names are. For example, <BLOCKQUOTE>, <BlockQuote>, and <blockquote> are equivalent, whereas & is different from &.

In a start tag, the element name must immediately follow the tag open delimiter "<".

2.2.3 Attributes

In a start tag, white space and attributes are allowed between the element name and the closing delimiter. An attribute typically consists of an attribute name, an equal sign, and a value (although some attributes may be just a value). White space is allowed around the equal sign.

The value of the attribute may be either:

- A string literal, delimited by single quotes or double quotes and not containing any occurrences of the delimiting character.
- A name token (a sequence of letters, digits, periods, or hyphens)

In this example, A is the element name, HREF is the attribute name, and http://host/dir/file.html is the attribute value:

```
<A HREF="http://host/dir/file.html">
```

> ***Note:*** *Some non-SGML implementations consider any occurrence of the ">" character to signal the end of a tag. For compatibility with such implementations, when ">" appears in an attribute value, it should be represented with a character entity reference, such as in: <IMG SRC="eq1.jpg" alt="a > b">*

To put quotes inside of quotes, the character entity " may be used, as in:

```
<IMG SRC="image.jpg" alt="First &quot;real&quot; example">
```

The length of an attribute value is limited to 1024 characters after replacing any character entity references.

Note: *Some non-SGML implementations allow any character except space or ">" in a name token. Attributes values must be quoted only if they don't satisfy the syntax for a name token.*

Attributes with a declared value of NAME, such as ISMAP and COMPACT, may be written using a minimized syntax. The markup:

```
<UL COMPACT="compact">
```

can be written using a minimized syntax:

```
<UL COMPACT>
```

Note: *Some non-SGML implementations only understand the minimized syntax.*

2.2.4 Entity References

SGML uses entity references, indicated by an ampersand (&) and immediately followed by a name and terminated by a semicolon (;), to represent a named substitution of data (the entity). HTML 2.0 only uses entity references to represent peculiar and special characters. The reference can be used in place of a character when the character itself would be misinterpreted as markup. The entity sets defined for use by HTML 2.0 documents are listed in Section 13.

The HTML DTD includes a character entity for each of the SGML markup characters, such that one may reference them by name if it is inconvenient to enter them directly:

GLYPH	NAMED	OCTET	CHARACTER NAME
&	&	&	Ampersand
"	"	"	Quotation mark
<	<	<	Less than
>	>	>	Greater than

To ensure that a sequence of data characters is not interpreted as markup, all occurrences of "<", ">", and "&" must be replaced by their character entity references.

Note: *There are SGML features, CDATA and RCDATA, to allow most "<", ">", and "&" characters to be entered without the use of entity references. Because these features tend to be used and implemented inconsistently, and because they require 8-bit characters to represent non-ASCII characters, they are not used in this version of the HTML DTD.*

2.2.5 Comments

To include comments in an HTML document that will be ignored by the interpreter, surround them with "<!--" and "-->". After the comment delimiter, all text up to the next occurrence of "-->" is ignored. Hence comments cannot be nested. White space is allowed between the closing "--" and ">", but not between the opening "<!" and "--".

For example:

```
<HEAD>
<TITLE>HTML Guide: Recommended Usage</TITLE>
<!-- $Id: Text.html,v 1.6 1994/04/25 17:33:48 connolly Exp$ -->
</HEAD>
```

Note: *Some historical HTML interpreters incorrectly consider any ">" character to be the termination of a comment.*

2.3 Example HTML Document

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<HTML>
<!-- Here's a good place to put a comment. -->
<HEAD>
<TITLE>Structural Example</TITLE>
</HEAD><BODY>
<H1>First Header</H1>
<P>This is a paragraph in the example HTML file. Keep in mind
that the title does not appear in the document text, but that
the header (defined by H1) does.</P>
<OL>
<LI>First item in an ordered list.
<LI>Second item in an ordered list.
  <UL COMPACT>
  <LI> Note that lists can be nested;
  <LI> Whitespace may be used to assist in reading the
       HTML source.
  </UL>
<LI>Third item in an ordered list.
</OL>

<P>This is an additional paragraph. Technically, end tags are
not required for paragraphs, although they are allowed. You can
include character highlighting in a paragraph. <EM>This sentence
of the paragraph is emphasized.</EM> Note that the &lt;/P&gt;
end tag has been omitted.
<P>
<IMG SRC ="triangle.xbm" alt="Warning:">
Be sure to read these <b>bold instructions</b>.
</BODY></HTML>
```

3. HTML as an Internet Media Type

3.1 text/html media type

This specification defines the Internet Media Type [7] (formerly referred to as the MIME Content Type [4]) called "text/html". The following is to be registered with IANA [8].

```
Media Type name:         text
Media subtype name:      html
Required parameters:     none
Optional parameters:     level, version, charset
Encoding considerations: any encoding is allowed
Security considerations: [Section 14]
```

The optional parameters are defined as follows:

Level
: The level parameter specifies the feature set used in the document. The level is an integer number, implying that any features of same or lower level may be present in the document. Levels 0, 1 and 2 are defined by this specification. Level 2 is the default.

Version
: To help avoid future compatibility problems, the version parameter may be used to give the version number of the specification to which the document conforms. The version number appears at the front of this document and within the public identifier of the HTML DTD. This specification defines version 2.0.

Charset
: The charset parameter (as defined in section 7.1.1 of RFC 1521 [4]) may be given to specify the encoding used to represent the HTML document as a sequence of octets.

3.2 Character Set Issues

An HTML interpreter must accept a stream of characters as input, and assign them to character classes used for markup recognition purposes. HTML 2.0 markup requires only the characters found in the US-ASCII character set [10]. All US-ASCII characters with no markup role, and all non-US-ASCII characters, should be treated as data. Such categorization should take place after decoding the data stream (i.e., not at the characer set encoding level, but rather at the character set level).

Normally, text/* media types specify a default value of US-ASCII for the charset parameter. However, for text/html, if the byte stream contains data that is not in the 7-bit US-ASCII set, the interpreter should assume a default charset of ISO-8859-1. Even if an HTML document is limited to a US-ASCII encoding, the mechanisms of character entity references (Section 6.3) can be used to encode the characters from ISO-8859-1.

Other values for the charset parameter may be defined by the transport mechanism (e.g., MIME and HTTP), but are not defined by this specification. Since the SGML declaration for HTML (supplied in Section 12.3) is only applicable to ISO-8859-1 and its subsets, a charset parameter that specifies a different character set must also imply a different SGML declaration. Therefore,

user agents may use the charset parameter value to select a different declaration, even though the mechanism for doing so is not defined by this specification. The intent, however, is that such a declaration be as identical as possible to that of Section 12.3, the only differences being those required to support the announced charset.

When the above conflicts with the SGML standard, the SGML standard may be ignored. Note, however, that not all HTML applications are capable of ignoring the SGML standard.

3.3 Undefined Tag and Attribute Names

An accepted networking principle is to be conservative in what one produces, and liberal in what one accepts. HTML user agents should be liberal except when validating code; user agents are encouraged to provide an option for validation of what they render, such that users may be notified of an invalid construct when such notification is desirable. HTML generators should generate strictly conforming HTML.

HTML user agents reading "text/html" documents and discovering tag or attribute names which they do not understand should behave as though the offending tags or attribute names do not exist. Any unknown start tag (including its entire attribute specification list) or end tag should be ignored; any content between matching unknown start and end tags should be treated as normal (i.e., as if those tags did not occur in the character stream). The entire attribute specification of an unknown attribute (i.e., the unknown attribute and its value, if any) should be ignored.

4. Document Structure Elements

To identify information as an HTML document conforming to this specification, each document should start with the prologue:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

Note: *If the body of a text/html body part does not begin with a document type declaration, an HTML user agent should infer the above document type declaration.*

HTML documents should also contain an <HTML> tag at the beginning (immediately after the prologue) and an </HTML> tag at the end of the file. Within those tags, an HTML document is organized as a head and a body, much like a memo or a mail message. Within the head, you can specify the title and other information about the document. Within the body, you can structure text into paragraphs and lists, as well as highlight phrases and create links, using HTML elements.

Note: *Technically, the start and end tags for HTML, Head, and Body elements are omissible; however, this is not recommended since the head/body structure allows an implementation to determine certain properties of a document, such as the title, without parsing the entire document.*

4.1 HTML Identifier

<HTML> ... </HTML> Level 0

The HTML identifier defines the document as containing HTML elements. It contains only the Head and Body elements.

4.2 Head

<HEAD> ... </HEAD> Level 0

The head of an HTML document is an unordered collection of information about the document. It requires the Title element between <HEAD> and </HEAD> tags:

```
<HEAD>
<TITLE>Introduction to HTML</TITLE>
</HEAD>
```

4.3 Body

<BODY> ... </BODY> Level 0

The Body element identifies the body component of an HTML document. Specifically, the body of a document may contain links, text, and formatting information within <BODY> and </BODY> tags.

5. Document Metainformation Elements

5.1 Title

<TITLE> ... </TITLE> Level 0

Every HTML document must contain a Title element. The title should identify the contents of the document in a global context, and may be used in history lists and as a label for the window displaying the document. Unlike headings, titles are not rendered in the text of a document itself.

The Title element must occur within the head of the document, and must not contain anchors, paragraph tags, or highlighting. Only one title is allowed in a document.

> ***Note:*** *The length of a title is not limited; however, long titles may be truncated in some applications. To minimize this possibility, titles should be fewer than 64 characters. Also keep in mind that a short title, such as Introduction, may be meaningless out of context. An example of a meaningful title might be "Introduction to HTML Elements."*

5.2 Base

<BASE> Level 0

The Base element allows the URL [2] of the document itself to be recorded in situations in which the document may be read out of context. URLs within the document may be in a "partial" form relative to this base address [5].

The Base element has one attribute, HREF, which identifies the absolute base URL.

5.3 Isindex

<ISINDEX> Level 0

The Isindex element tells the interpreter that the document is an index. This means that the reader may request a keyword search on the resource by adding a question mark to the end of the document address, followed by a list of keywords separated by plus signs.

The Isindex element is usually generated by the network server from which the document was obtained via a URI. The server must have a search engine that supports this feature for the resource. If the document URI is unknown to the interpreter, <isindex> must be ignored.

5.4 Link

<LINK> Level 0

The Link element indicates a relationship between the document and some other object. A document may have any number of Link elements.

The Link element is empty (does not have a closing tag), but takes the same attributes as the Anchor element.

Typical uses are to indicate authorship, related indexes and glossaries, older or more recent versions, etc. Links can indicate a static tree structure in which the document was authored by pointing to a "parent" and "next" and "previous" document, for example.

Servers may also allow links to be added by those who do not have the right to alter the body of a document.

5.5 Meta

<META> Level 0

The META element is used within the HEAD element to embed document metainformation not defined by other HTML elements. META elements can be extracted by servers and/or clients for use in identifying, indexing, and cataloging specialized document metainformation.

Although it is generally preferable to use named elements which have well-defined semantics for each type of metainformation (e.g. TITLE), the META element is provided for situations where strict SGML parsing is necessary and the local DTD is not extensible. HTML interpreters may use the META element's content if they recognize and understand the semantics identified by the NAME or HTTP-EQUIV attributes, and may treat the content as metainformation (and not render it) even when they do not recognize the name.

In addition, HTTP servers may wish to read the content of the document HEAD to generate header fields corresponding to any elements defining a value for the attribute HTTP-EQUIV. Note, however, that the method by which the server extracts document metainformation is not part of this specification, nor can it be assumed by authors that any given server will be capable of extracting it. The META element only provides an extensible mechanism for identifying and embedding document metainformation — how it may be used is up to the individual server implementation and the HTML interpreter.

Attributes of the META element:

HTTP-EQUIV

This attribute binds the element to an HTTP header field. It means that if you know the semantics of the HTTP header field named by this attribute, then you can process the contents based on a well-defined syntactic mapping, whether or not your DTD tells you anything about it. HTTP header field names are not case sensitive. If not present, the attribute NAME should be used to identify this metainformation and the content should not be used within an HTTP response header.

NAME
Metainformation name. If the NAME attribute is not present, the name can be assumed to be equal to the value of HTTP-EQUIV.

CONTENT
The metainformation content to be associated with the given name. If multiple META elements are provided with the same name, their combined contents—concatenated as a comma-separated list—is the value associated with that name.

Examples

If the document contains:

```
<META HTTP-EQUIV="Expires"
      CONTENT="Tue, 04 Dec 1993 21:29:02 GMT">
<meta http-equiv="Keywords" CONTENT="Fred, Barney">
<META HTTP-EQUIV="Reply-to"
      content="fielding@ics.uci.edu (Roy Fielding)">
```

then the server (if so configured) may include the following headers:

```
Expires: Tue, 04 Dec 1993 21:29:02 GMT
Keywords: Fred, Barney
Reply-to: fielding@ics.uci.edu (Roy Fielding)
```

as part of the HTTP response to a GET or HEAD request for that document.

When the HTTP-EQUIV attribute is not present, the server should not generate an HTTP response header for the metainformation; e.g.,

```
<META NAME="IndexType" CONTENT="Service">
```

would never generate an HTTP response header, but would still allow HTML interpreters to identify and make use of that metainformation.

The Meta element should never be used to define information that should be associated with an existing HTML element. An example of an inappropriate use of the Meta element is:

```
<META NAME="Title" CONTENT="The Etymology of Dunsel">
```

Do not name an HTTP-EQUIV equal to a response header that should normally only be generated by the HTTP server. Example names that are inappropriate include "Server", "Date", and "Last-modified" — the exact list of inappropriate names is dependent on the particular server implementation. We recommend that servers ignore any META elements which specify HTTP-equivalents which are equal (case-insensitively) to their own reserved response headers.

5.6 Nextid

<NEXTID> Level 0

The Nextid element is a parameter read and generated by text editing software to create unique identifiers. This tag takes a single attribute which is the next document-wide alpha-numeric identifier to be allocated of the form z123:

```
<NEXTID N=Z27>
```

When modifying a document, existing anchor identifiers should not be reused, as these identifiers may be referenced by other documents. Human writers of HTML usually use mnemonic alphabetical identifiers.

HTML interpreters may ignore the Nextid element. Support for the Nextid element does not impact HTML interpreters in any way.

6. Data Characters

The characters between HTML tags represent text. An HTML document (including tags and text) is encoded using the coded character set specified by the "charset" parameter of the "text/html" media type. For levels defined in this specification, the "charset" parameter is restricted to US-ASCII [10] or ISO-8859-1 [11]. ISO-8859-1 encodes a set of characters known as Latin Alphabet No. 1, or simply Latin-1. Latin-1 includes characters from most Western European languages, as well as a number of control characters. Latin-1 also includes a non-breaking space, a soft hyphen indicator, 93 graphical characters, 8 unassigned characters, and 25 control characters.

Use the non-breaking space and soft hyphen indicator characters is discouraged because they are not recognized and interpreted by all HTML interpreters.

Because certain characters will be interpreted as markup, they must be represented by entity references. HTML provides character entity references to facilitate the entry and interpretation of characters by name or by numerical position (octet).

6.1 Special Characters

Certain data characters have special meaning in HTML documents. There are two printing characters which may be interpreted by an HTML application to have an effect of the format of the text:

Space

- Interpreted as a word space (place where a line can be broken) in all contexts except the Preformatted Text element.
- Interpreted as a nonbreaking space within the Preformatted Text element.

Hyphen

- Iterpreted as a hyphen glyph in all contexts
- Interpreted as a potential word space by hyphenation engine

6.2 Control Characters

Control characters are non-printable characters that are typically used for communication and device control, as format effectors, and as information separators. There are 58 character positions occupied by control characters.

In SGML applications, the use of control characters is limited in order to maximize the chance of successful interchange over heterogenous networks and operating systems. In HTML, only three control characters are used: Horizontal Tab (HT, encoded as 9 decimal in US-ASCII and ISO-8859-1), Carriage Return, and Line Feed.

Horizontal Tab is interpreted as a word space in all contexts except preformatted text. Within preformatted text, the tab should be interpreted to shift the horizontal column position to the next position which is a multiple of 8 on the same line; that is, col := (col+8) mod 8.

Carriage Return and Line Feed are conventionally used to represent end of line. For Internet Media Types defined as "text/*", the sequence CR LF is used to represent an end of line. In practice, text/html documents are frequently represented and transmitted using an end of line convention that depends on the conventions of the source of the document; frequently, that representation consists of CR only, LF only, or CR LF combination. In HTML, end of line in any of its variations is interpreted as a word space in all contexts except preformatted text. Within preformatted text, HTML interpreters should expect to treat any of the three common representations of end-of-line as starting a new line.

6.3 Character Entities

Two reasons for using a character entity reference:

- the keyboard does not provide a key for the character, such as on U.S. keyboards which do not provide European characters
- the character may be interpreted as SGML coding, such as the ampersand (&), double quotes ("), the lesser (<) and greater (>) characters

A character entity is represented in an HTML document as an SGML entity whose name is defined in the HTML DTD.

6.3.1 Character Name References

Most of the Latin alphabet No. 1 set of printing characters may be represented within the text of an HTML document by a character entity. See Section 13.1 for a list of the characters, names, input syntax, and descriptions for numeric and special graphic characters. See Section 13.2 for the SGML entity definitions of "Added Latin 1 for HTML".

```
Kurt G&ouml;del was a famous logician and mathematician.
```

6.3.2 Character Octet References

It is possible to explicitly reference the printing characters of the ISO-8859-1 character encoding using a character octet reference. See Section 13.3 for a list of the characters, their names and input syntax.

Character octet references are represented in an HTML document as SGML entities whose name is number sign (#) followed by a numeral from 32-126 and 161-255. The HTML DTD includes a numeric character for each of the printing characters of the ISO-8859-1 encoding, so that one may reference them by number if it is inconvenient to enter them directly.

The character octet references are not dependent on the character set encoding of the document. For example, "×" always represents the ISO-8859-1 multiply sign, even when the document's declared character set is other than ISO-8859-1.

7. Data Elements

7.1 Line Break

 Level 0

The Line Break element specifies that a new line must be started at the given point. A new line indents the same as that of line-wrapped text.

Example of use:

```
<P> Pease porridge hot<BR>
Pease porridge cold<BR>
Pease porridge in the pot<BR>
Nine days old.
```

7.2 Horizontal Rule

<HR> Level 0

A Horizontal Rule element is a divider between sections of text such as a full width horizontal rule or equivalent graphic.

Example of use:

```
<HR>
<ADDRESS>February 8, 1995, CERN</ADDRESS>
</BODY>
```

7.3 Image

<IMG> Level 0

The Image element is used to incorporate in-line graphics (typically icons or small graphics) into an HTML document. This element cannot be used for embedding other HTML text.

HTML interpreters that cannot render in-line images ignore the Image element unless it contains the ALT attribute. Note that some HTML interpreters can render linked graphics but not in-line graphics. If a graphic is essential, you may want to create a link to it rather than to put it in-line. If the graphic is not essential, then the Image element is appropriate.

The Image element, which is empty (no closing tag), has these attributes:

ALIGN
: The ALIGN attribute accepts the values TOP or MIDDLE or BOTTOM, which specifies if the following line of text is aligned with the top, middle, or bottom of the graphic.

ALT
: Optional text as an alternative to the graphic for rendering in non-graphical environments. Alternate text should be provided whenever the graphic is not rendered. Alternate text is mandatory for Level 0 documents. Example of use:

```
<IMG SRC="triangle.xbm" ALT="Warning:"> Be sure to read these
instructions.
```

ISMAP

The ISMAP (is map) attribute identifies an image as an image map. Image maps are graphics in which certain regions are mapped to URLs. By clicking on different regions, different resources can be accessed from the same graphic. Example of use:

```
<A HREF="http://machine/htbin/imagemap/sample">
<IMG SRC="sample.xbm" ISMAP>
</A>
```

SRC

The value of the SRC attribute is the URL of the document to be embedded; only images can be embedded, not HTML text. Its syntax is the same as that of the HREF attribute of the <A> tag. SRC is mandatory. Image elements are allowed within anchors.

Example of use:

```
<IMG SRC="triangle.xbm">Be sure to read these instructions.
```

8. Character Format Elements

Character format elements are used to specify either the logical meaning or the physical appearance of marked text without causing a paragraph break. Like most other elements, character format elements include both opening and closing tags. Only the characters between the tags are affected:

```
This is <EM>emphasized</EM> text.
```

Character format tags may be ignored by minimal HTML applications.

Character format tags are interpreted from left to right as they appear in the flow of text. Level 1 interpreters must render highlighted text distinctly from plain text. Additionally, EM content must be rendered as distinct from STRONG content, and B content must rendered as distinct from I content.

Character format elements may be nested within the content of other character format elements; however, HTML interpreters are not required to render nested character format elements distinctly from non-nested elements:

```
plain <B>bold <I>italic</I></B> may the rendered the same as
plain <B>bold </B><I>italic</I>
```

8.1 Semantic Format Elements

Note that typical renderings for semantic format elements vary between applications. If a specific rendering is necessary — for example, when referring to a specific text attribute as in "The italic parts are mandatory" — a physical formating element can be used to ensure that the intended rendered is used where possible.

Note that different sematic elements may be rendered in the same way.

8.1.1 Citation

<CITE>...</CITE> Level 1

The Citation element specifies a citation, typically rendered as italics.

8.1.2 Code

<CODE> ... </CODE> Level 1

The Code element indicates an example of code, typically rendered in a monospaced font. This should not be confused with the Preformatted Text element.

8.1.3 Emphasis

<EM> ... </EM> Level 1

The Emphasis element indicates typographic emphasis, typically rendered as italics.

8.1.4 Keyboard

<KBD> ... </KBD> Level 1

The Keyboard element indicates text typed by a user, typically rendered in a monospaced font. This is commonly used in instruction manuals.

8.1.5 Sample

<SAMP> ... </SAMP> Level 1

The Sample element indicates a sequence of literal characters, typically rendered in a monospaced font.

8.1.6 Strong

<STRONG> ... </STRONG> Level 1

The Strong element indicates strong typographic emphasis, typically rendered in bold.

8.1.7 Variable

<VAR> ... </VAR> Level 1

The Variable element indicates a variable name, typically rendered as italic.

8.2 Physical Format Elements

Physical format elements are used to specify the format of marked text.

8.2.1 Bold

<B> ... </B> Level 1

The Bold element specifies that the text should be rendered in boldface, where available. Otherwise, an alternative mapping is allowed.

8.2.2 Italic

<I> ... </I> Level 1

The Italic element specifies that the text should be rendered in an italic font, where available. Otherwise, an alternative mapping is allowed.

8.2.3 Teletype

<TT> ... </TT> Level 1

The Teletype element specifies that the text should be rendered in a fixed-width typewriter font.

9. Hypertext Elements

9.1 Anchor

<A> ... </A> Level 0

An anchor is a marked section of text that is the start and/or destination of a hypertext link. Anchor elements are defined by the <A> tag. The <A> tag accepts several attributes; at least one of the NAME and HREF attributes is required.

Attributes of the <A> tag:

9.1.1 HREF

If the HREF attribute is present, the text between the opening and closing anchor tags becomes hypertext. If this hypertext is selected by readers, they are moved to another document, or to a different location in the current document, whose network address is defined by the value of the HREF attribute.

Example:

```
See <A HREF="http://www.hal.com/">HaL</A>'s information for more
details.
```

In this example, selecting "HaL" takes the reader to a document at http://www.hal.com. The format of the network address is specified in the URI specification for print readers.

With the HREF attribute, the form HREF="#identifier" can refer to another anchor in the same document.

Example:

```
The <A HREF="#glossary">glossary</A> defines terms used in this
document.
```

In this example, selecting "glossary" takes the reader to another anchor (i.e., <A NAME="glossary">Glossary</A>) in the same document. The NAME attribute is described below. If the anchor is in another document, the HREF attribute may be relative to the document's address or the specified base address (see Section 5.2).

9.1.2 NAME

If present, the NAME attribute allows the anchor to be the target of a link. The value of the NAME attribute is an identifier for the anchor. Identifiers are arbitrary strings but must be unique within the HTML document.

Example of use:

```
<A NAME="coffee">Coffee</A> is an example of ...
... An example of this is <A HREF="#coffee">coffee</A>.
```

Another document can then make a reference explicitly to this anchor by putting the identifier after the address, separated by a hash sign:

```
<A HREF="drinks.html#coffee">
```

9.1.3 TITLE

The TITLE attribute is informational only. If present, the TITLE attribute should provide the title of the document whose address is given by the HREF attribute. The TITLE attribute is useful for at least two reasons. The HTML interpreter may display the title of the document prior to retrieving it, for example, as a margin note or on a small box while the mouse is over the anchor, or while the document is being loaded. Another reason is that documents that are not marked up text, such as graphics, plain text and Gopher menus, do not have titles. The TITLE attribute can be used to provide a title to such documents. When using the TITLE attribute, the title should be valid and unique for the destination document.

9.1.4 REL

The REL attribute gives the relationship(s) described by the hypertext link from the anchor to the target. The value is a whitespace-separated list of relationship names. Relationship names and their semantics will be registered by the W3 Consortium. The default relationship is void. The REL attribute is only used when the HREF attribute is present.

9.1.5 REV

The REV attribute is the same as the REL attribute, but the semantics of the link type are in the reverse direction. A link from A to B with REL="X" expresses the same relationship as a link from B to A with REV="X". An anchor may have both REL and REV attributes.

9.1.6 URN

If present, the URN attribute specifies a uniform resource name (URN) for a target document. The format of URNs is under discussion (1995) by various working groups of the Internet Engineering Task Force.

9.1.7 METHODS

The METHODS attributes of anchors and links provide information about the functions that the user may perform on an object. These are more accurately given by the HTTP protocol when it is used, but it may, for similar reasons as for the TITLE attribute, be useful to include the information in advance in the link. For example, the HTML interpreter may chose a different rendering as a function of the methods allowed; for example, something that is searchable may get a different icon.

The value of the METHODS attribute is a whitespace-separated list of HTTP methods supported by the object for public use.

10. Block Structuring Elements

The following elements may be included in the body of an HTML document:

10.1 Paragraph

<P> ... </P> Level 0

The Paragraph element indicates a paragraph. The exact indentation, leading space, etc. of a paragraph is not defined and may be a function of other tags, style sheets, etc.

Typically, paragraphs are surrounded by a vertical space of one line or half a line. This is typically not the case within the Address element and is never the case within the Preformatted Text element. With some HTML interpreters, the first line in a paragraph is indented.

Example of use:

```
<H1>This Heading Precedes the Paragraph</H1>
<P>This is the text of the first paragraph.
<P>This is the text of the second paragraph. Although you do not
need to start paragraphs on new lines, maintaining this
convention facilitates document maintenance.</P>
<P>This is the text of a third paragraph.</P>
```

10.2 Preformatted Text

<PRE> ... </PRE> Level 0

The Preformatted Text element presents blocks of text in fixed-width font, and so is suitable for text that has been formatted on screen.

The <PRE> tag may be used with the optional WIDTH attribute. The WIDTH attribute specifies the maximum number of characters for a line and allows the HTML interpreter to select a suitable font and indentation. If the WIDTH attribute is not present, a width of 80 characters is assumed. Where the WIDTH attribute is supported, widths of 40, 80 and 132 characters should be presented optimally, with other widths being rounded up.

Within preformatted text:

- Line breaks within the text are rendered as a move to the beginning of the next line.
- The <P> tag should not be used. If found, it should be rendered as a move to the beginning of the next line.
- Anchor elements and character highlighting elements may be used.
- Elements that define paragraph formatting (headings, address, etc.) must not be used.
- The horizontal tab character (encoded in US-ASCII and ISO-8859-1 as decimal 9) must be interpreted as the smallest positive nonzero number of spaces which will leave the number of characters so far on the line as a multiple of 8. Its use is not recommended however.

Note: *References to the "beginning of a new line" do not imply that the renderer is forbidden from using a constant left indent for rendering preformatted text. The left indent may be constrained by the width required.*

Example of use:

```
<PRE WIDTH="80">
This is an example line.
</PRE>
```

Note: *Within a Preformatted Text element, the constraint that the rendering must be on a fixed horizontal character pitch may limit or prevent the ability of the HTML interpreter to faithfully render character formatting elements.*

10.3 Address

<ADDRESS> ... </ADDRESS> Level 0

The Address element specifies such information as address, signature and authorship, often at the top or bottom of a document.

Typically, an Address is rendered in an italic typeface and may be indented. The Address element implies a paragraph break before and after.

Example of use:

```
<ADDRESS>
Newsletter editor<BR>
J.R. Brown<BR>
JimquickPost News, Jumquick, CT 01234<BR>
Tel (123) 456 7890
</ADDRESS>
```

10.4 Blockquote

<BLOCKQUOTE> ... </BLOCKQUOTE> Level 0

The Blockquote element is used to contain text quoted from another source.

A typical rendering might be a slight extra left and right indent, and/or italic font. The Blockquote element causes a paragraph break, and typically provides space above and below the quote.

Single-font rendition may reflect the quotation style of Internet mail by putting a vertical line of graphic characters, such as the greater than symbol (>), in the left margin.

Example of use:

```
I think the poem ends
<BLOCKQUOTE>
<P>Soft you now, the fair Ophelia. Nymph, in thy orisons, be all
my sins remembered.
</BLOCKQUOTE>
but I am not sure.
```

10.5 Headings

<H1> ... </H1> through <H6> ... </H6> Level 0

HTML defines six levels of heading. A Heading element implies all the font changes, paragraph breaks before and after, and white space necessary to render the heading.

The highest level of headings is H1, followed by H2 ... H6.

Example of use:

```
<H1>This is a heading</H1>
Here is some text
<H2>Second level heading</H2>
Here is some more text.
```

The rendering of headings is determined by the HTML interpreter, but typical renderings are:

<H1> ... </H1>
Bold, very-large font, centered. One or two blank lines above and below.

<H2> ... </H2>
Bold, large font, flush-left. One or two blank lines above and below.

<H3> ... </H3>
Italic, large font, slightly indented from the left margin. One or two blank lines above and below.

<H4> ... </H4>
Bold, normal font, indented more than H3. One blank line above and below.

<H5> ... </H5>
Italic, normal font, indented as H4. One blank line above.

<H6> ... </H6>
Bold, indented same as normal text, more than H5. One blank line above.

Although heading levels can be skipped (for example, from H1 to H3), this practice is discouraged as skipping heading levels may produce unpredictable results when generating other representations from HTML.

10.6 List Elements

HTML supports several types of lists, all of which may be nested.

10.6.1 Definition List

<DL> ... </DL> Level 0

A definition list is a list of terms and corresponding definitions. Definition lists are typically formatted with the term flush-left and the definition, formatted paragraph style, indented after the term.

Example of use:

```
<DL>
<DT>Term<DD>This is the definition of the first term.
<DT>Term<DD>This is the definition of the second term.
</DL>
```

If the DT term does not fit in the DT column (one third of the display area), it may be extended across the page with the DD section moved to the next line, or it may be wrapped onto successive lines of the left hand column.

Single occurrences of a <DT> tag without a subsequent <DD> tag are allowed, and have the same significance as if the <DD> tag had been present with no text.

The opening list tag must be <DL> and must be immediately followed by the first term (<DT>).

The definition list type can take the COMPACT attribute, which suggests that a compact rendering be used, because the list items are small and/or the entire list is large.

Unless you provide the COMPACT attribute, the HTML interpreter may leave white space between successive DT, DD pairs. The COMPACT attribute may also reduce the width of the left-hand (DT) column.

If using the COMPACT attribute, the opening list tag must be <DL COMPACT>, which must be immediately followed by the first <DT> tag:

```
<DL COMPACT>
<DT>Term<DD>This is the first definition in compact format.
<DT>Term<DD>This is the second definition in compact format.
</DL>
```

10.6.2 Directory List

<DIR> ... </DIR> Level 0

A Directory List element is used to present a list of items containing up to 20 characters each. Items in a directory list may be arranged in columns, typically 24 characters wide. If the HTML interpreter can optimize the column width as function of the widths of individual elements, so much the better.

A directory list must begin with the <DIR> tag which is immediately followed by a <LI> (list item) tag:

```
<DIR>
<LI>A-H<LI>I-M
<LI>M-R<LI>S-Z
</DIR>
```

10.6.3 Menu List

<MENU> ... </MENU> Level 0

A menu list is a list of items with typically one line per item. The menu list style is more compact than the style of an unordered list.

A menu list must begin with a <MENU> tag which is immediately followed by a <LI> (list item) tag:

```
<MENU>
<LI>First item in the list.
<LI>Second item in the list.
<LI>Third item in the list.
</MENU>
```

10.6.4 Ordered List

<OL> ... </OL> Level 0

The Ordered List element is used to present a numbered list of items, sorted by sequence or order of importance.

An ordered list must begin with the <OL> tag which is immediately followed by a <LI> (list item) tag:

```
<OL>
<LI>Click the Web button to open the Open the URL window.
<LI>Enter the URL number in the text field of the Open URL
window. The Web document you specified is displayed.
<LI>Click highlighted text to move from one link to another.
</OL>
```

The Ordered List element can take the COMPACT attribute, which suggests that a compact rendering be used.

10.6.5 Unordered List

<UL> ... </UL> Level 0

The Unordered List element is used to present a list of items which is typically separated by white space and/or marked by bullets.

An unordered list must begin with the <UL> tag which is immediately followed by a <LI> (list item) tag:

```
<UL>
<LI>First list item
<LI>Second list item
<LI>Third list item
</UL>
```

11. Form-based Input Elements

Forms are created by placing input fields within paragraphs, preformatted/literal text, and lists. This gives considerable flexibility in designing the layout of forms.

The following elements are used to create forms:

FORM
A form within a document.

INPUT
One input field.

OPTION
One option within a Select element.

SELECT
A selection from a finite set of options.

TEXTAREA
A multi-line input field.

Each variable field is defined by an Input, Textarea, or Option element and must have an NAME attribute to identify its value in the data returned when the form is submitted.

Example of use (a questionnaire form):

```
<H1>Sample Questionnaire</H1>
<P>Please fill out this questionnaire:
<FORM METHOD="POST" ACTION="http://www.w3.org/sample">
<P>Your name: <INPUT NAME="name" size="48">
<P>Male <INPUT NAME="gender" TYPE=RADIO VALUE="male">
<P>Female <INPUT NAME="gender" TYPE=RADIO VALUE="female">
<P>Number in family: <INPUT NAME="family" TYPE=text>
<P>Cities in which you maintain a residence:
<UL>
<LI>Kent <INPUT NAME="city" TYPE=checkbox VALUE="kent">
<LI>Miami <INPUT NAME="city" TYPE=checkbox VALUE="miami">
<LI>Other <TEXTAREA NAME="other" cols=48 rows=4></textarea>
</UL>
Nickname: <INPUT NAME="nickname" SIZE="42">
<P>Thank you for responding to this questionnaire.
<P><INPUT TYPE=SUBMIT> <INPUT TYPE=RESET>
</FORM>
```

In the example above, the <P> and <UL> tags have been used to lay out the text and input fields. The HTML interpreter is responsible for handling which field will currently get keyboard input.

Many platforms have existing conventions for forms, for example, using Tab and Shift keys to move the keyboard focus forwards and backwards between fields, and using the Enter key to submit the form. In the example, the SUBMIT and RESET buttons are specified explicitly with special purpose fields. The SUBMIT button is used to e-mail the form or send its contents to

the server as specified by the ACTION attribute, while RESET resets the fields to their initial values. When the form consists of a single text field, it may be appropriate to leave such buttons out and rely on the Enter key.

The Input element is used for a large variety of types of input fields.

To let users enter more than one line of text, use the Textarea element.

The radio button and checkbox types of input field can be used to specify multiple choice forms in which every alternative is visible as part of the form. An alternative is to use the Select element which is typically rendered in a more compact fashion as a pull down combo list.

11.1 Form

<FORM> ... </FORM> Level 2

The Form element is used to delimit a data input form. There can be several forms in a single document, but the Form element can't be nested.

The ACTION attribute is a URL specifying the location to which the contents of the form is submitted to elicit a response. If the ACTION attribute is missing, the URL of the document itself is assumed. The way data is submitted varies with the access protocol of the URL, and with the values of the METHOD and ENCTYPE attributes.

In general:

- the METHOD attribute selects variations in the protocol.
- the ENCTYPE attribute specifies the format of the submitted data in case the protocol does not impose a format itself.

The Level 2 specification defines and requires support for the HTTP access protocol only.

When the ACTION attribute is set to an HTTP URL, the METHOD attribute must be set to an HTTP method [3]. The default method is GET, although for many applications the POST method is preferred. With the POST method, the ENCTYPE attribute is a media type specifying the format of the posted data; the default is "application/x-www-form-urlencoded".

The submitted contents of the form logically consist of name/value pairs. The names are usually equal to the NAME attributes of the various interactive elements in the form.

> ***Note:*** *The names are not guaranteed to be unique keys, nor are the names of form elements required to be distinct. The values encode the user's input to the corresponding interactive elements. Fields with null values may be omitted from the returned list of name/value pairs, whereas those with non-null values should be included (even if the value was not altered by the user). In particular, unselected radio buttons and checkboxes should be excluded from the contents list.*

11.2 Input

<INPUT> Level 2

The Input element represents a field whose contents may be edited by the user.

Attributes of the Input element:

ALIGN
Vertical alignment of the image. For use only with TYPE=IMAGE. The possible values are exactly the same as for the ALIGN attribute of the image element.

CHECKED
Indicates that a checkbox or radio button is selected. Unselected checkboxes and radio buttons do not return name/value pairs when the form is submitted.

MAXLENGTH
Indicates the maximum number of characters that can be entered into a text field. This can be greater than specified by the SIZE attribute, in which case the field will scroll appropriately. The default number of characters is unlimited.

NAME
Symbolic name used when transferring the form's contents. The NAME attribute is required for most input types and is normally used to provide a unique identifier for a field, or for a logically related group of fields.

SIZE
Specifies the size or precision of the field according to its type. For example, to specify a field with a visible width of 24 characters:

```
INPUT TYPE=text SIZE="24"
```

SRC
A URL or URN specifying an image. For use only with TYPE=IMAGE.

TYPE
Defines the type of data the field accepts. Defaults to free text. Several types of fields can be defined with the type attribute:

CHECKBOX
Used for simple Boolean attributes, or for attributes that can take multiple values at the same time. The latter is represented by a number of checkbox fields each of which has the same name. Each selected checkbox generates a separate name/value pair in the submitted data, even if this results in duplicate names. The default value for checkboxes is "on".

HIDDEN
No field is presented to the user, but the content of the field is sent with the submitted form. This value may be used to transmit state information about client/server interaction.

IMAGE
An image field upon which you can click with a pointing device, causing the form to be immediately submitted. The coordinates of the selected point are measured in pixel units from the upper-left corner of the image, and are returned (along with the other contents of the form) in two name/value pairs. The x-coordinate is submitted under the name of the field with ".x" appended, and the y-coordinate is submitted under the name of the field with ".y" appended. Any VALUE attribute is ignored. The image itself is specified by the SRC attribute, exactly as for the Image element.

Note: *In a future version of the HTML specification, the IMAGE functionality may be folded into an enhanced SUBMIT field.*

PASSWORD
The same as the TEXT attribute, except that text is not displayed as it is entered.

RADIO
Used for attributes that accept a single value from a set of alternatives. Each radio button field in the group should be given the same name. Only the selected radio button in the group generates a name/value pair in the submitted data. Radio buttons require an explicit VALUE attribute.

RESET
A button that when pressed resets the form's fields to their specified initial values. The label to be displayed on the button may be specified just as for the SUBMIT button.

SUBMIT
A button that when pressed submits the form. You can use the VALUE attribute to provide a non-editable label to be displayed on the button. The default label is application-specific. If a SUBMIT button is pressed in order to submit the form, and that button has a NAME attribute specified, then that button contributes a name/value pair to the submitted data. Otherwise, a SUBMIT button makes no contribution to the submitted data.

TEXT
Used for a single line text entry fields. Use in conjunction with the SIZE and MAXLENGTH attributes. Use the Textarea element for text fields which can accept multiple lines.

VALUE
The initial displayed value of the field, if it displays a textual or numerical value; or the value to be returned when the field is selected, if it displays a Boolean value. This attribute is required for radio buttons.

11.3 Option

<OPTION> Level 2

The Option element can only occur within a Select element. It represents one choice, and can take these attributes:

SELECTED
Indicates that this option is initially selected.

VALUE
When present indicates the value to be returned if this option is chosen. The returned value defaults to the contents of the Option element.

The contents of the Option element is presented to the user to represent the option. It is used as a returned value if the VALUE attribute is not present.

11.4 Select

<SELECT NAME=... > ... </SELECT> Level 2

The Select element allows the user to chose one of a set of alternatives described by textual labels. Every alternative is represented by the Option element. Attributes are:

MULTIPLE
The MULTIPLE attribute is needed when users are allowed to make several selections, e.g. <SELECT MULTIPLE>.

NAME
Specifies the name that will submitted as a name/value pair.

SIZE
Specifies the number of visible items. If this is greater than one, then the resulting form control will be a list.

The Select element is typically rendered as a pull down or pop-up list. For example:

```
<SELECT NAME="flavor">
<OPTION>Vanilla
<OPTION>Strawberry
<OPTION>Rum and Raisin
<OPTION>Peach and Orange
</SELECT>
```

If no option is initially marked as selected, then the first item listed is selected.

11.5 Text Area

<TEXTAREA> ... </TEXTAREA> Level 2

The Textarea element lets users enter more than one line of text. For example:

```
<TEXTAREA NAME="address" ROWS=64 COLS=6>
HaL Computer Systems
1315 Dell Avenue
Campbell, California 95008
</TEXTAREA>
```

The text up to the end tag (</TEXTAREA>) is used to initialize the field's value. This end tag is always required even if the field is initially blank. When submitting a form, lines in a TEXTAREA should be terminated using CRLF.

In a typical rendering, the ROWS and COLS attributes determine the visible dimension of the field in characters. The field is rendered in a fixed-width font. HTML interpreters should allow text to extend beyond these limits by scrolling as needed.

> ***Note:*** *In the initial design for forms, multi-line text fields were supported by the Input element with TYPE=TEXT. Unfortunately, this causes problems for fields with long text values. SGML's default (Reference Quantity Set) limits the length of attribute literals to only 240 characters. The HTML 2.0 SGML declaration increases the limit to 1024 characters.*

12. HTML Document Type Definitions

12.1 Deprecated and Recommended Sections

Optional "deprecated" and "recommended" sections are used in the HTML DTD. Conformance with this specification is defined with these sections disabled. In the liberal spirit of Section 3.3, user agents reading HTML documents should accept syntax corresponding to the specification with "deprecated" turned on. On the other hand, applications generating HTML should, in the conservative spirit, generate documents that conform to the specification with the "recommended" sections turned on.

12.2 Sample SGML Open Entity Catalog for HTML

The SGML standard describes an "entity manager" as the portion or component of an SGML system that maps SGML entities into the actual storage model (e.g., the file system). The standard itself does not define a particular mapping methodology or notation.

To assist the interoperability among various SGML tools and systems, the SGML Open consortium has passed a technical resolution that defines a format for an application-independent entity catalog that maps external identifiers and/or entity names to file names.

Each entry in the catalog associates a storage object identifier (such as a file name) with information about the external entity that appears in the SGML document. In addition to entries that associate public identifiers, a catalog entry can associate an entity name with a storage object indentifier. For example, the following are possible catalog entries:

```
PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN" "iso-lat1.gml"
PUBLIC "-//ACME DTD Writers//DTD General Report//EN" report.dtd
ENTITY "graph1" "graphics\graph1.cgm"
```

In particular, the following shows entries relevant to HTML.

```
-- catalog: SGML Open style entity catalog for HTML --

-- Ways to refer to Level 2: most general to most specific --
PUBLIC  "-//IETF//DTD HTML//EN"                 html.dtd
PUBLIC  "-//IETF//DTD HTML 2.0//EN"             html.dtd
PUBLIC  "-//IETF//DTD HTML Level 2//EN"         html.dtd
PUBLIC  "-//IETF//DTD HTML 2.0 Level 2//EN"     html.dtd

-- Ways to refer to Level 1: most general to most specific --
PUBLIC  "-//IETF//DTD HTML Level 1//EN"         html-1.dtd
PUBLIC  "-//IETF//DTD HTML 2.0 Level 1//EN"     html-1.dtd

-- Ways to refer to Level 0: most general to most specific --
PUBLIC  "-//IETF//DTD HTML Level 0//EN"         html-0.dtd
PUBLIC  "-//IETF//DTD HTML 2.0 Level 0//EN"     html-0.dtd

-- ISO latin 1 entity set for HTML --
PUBLIC  "ISO 8879-1986//ENTITIES Added Latin 1//EN//HTML"
        ISOlat1.sgml
```

12.3 SGML Declaration for HTML

This is the SGML Declaration for HyperText Markup Language (HTML) as used by the World Wide Web (WWW) application:

```
<!SGML  "ISO 8879:1986"
--
        SGML Declaration for HyperText Markup Language (HTML).
--
CHARSET
      BASESET  "ISO 646:1983//CHARSET
                International Reference Version
                (IRV)//ESC 2/5 4/0"
      DESCSET  0   9   UNUSED
               9   2   9
               11  2   UNUSED
               13  1   13
               14  18  UNUSED
               32  95  32
               127 1   UNUSED
      BASESET  "ISO Registration Number 100//CHARSET
                ECMA-94 Right Part of
                Latin Alphabet Nr. 1//ESC 2/13 4/1"

      DESCSET  128  32   UNUSED
               160  96    32

CAPACITY       SGMLREF
               TOTALCAP        150000
               GRPCAP          150000

SCOPE  DOCUMENT
SYNTAX
       SHUNCHAR CONTROLS 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
                17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 127
       BASESET  "ISO 646:1983//CHARSET
                 International Reference Version
                 (IRV)//ESC 2/5 4/0"
       DESCSET  0 128 0
       FUNCTION
                RE          13
                RS          10
                SPACE       32
                TAB SEPCHAR  9


       NAMING   LCNMSTRT ""
                UCNMSTRT ""
                LCNMCHAR ".-"
                UCNMCHAR ".-"
                NAMECASE GENERAL YES
                         ENTITY  NO
       DELIM    GENERAL  SGMLREF
                SHORTREF SGMLREF
```

```
        NAMES    SGMLREF
        QUANTITY SGMLREF
                 ATTSPLEN 2100
                 LITLEN   1024
                 NAMELEN  72    -- somewhat arbitrary;
                                   taken from Internet line
                                   length conventions --
                 PILEN    1024
                 TAGLEN   2100
                 GRPGTCNT 150
                 GRPCNT   64

    FEATURES
      MINIMIZE
        DATATAG  NO
        OMITTAG  YES
        RANK     NO
        SHORTTAG YES
      LINK
        SIMPLE   NO
        IMPLICIT NO
        EXPLICIT NO
      OTHER
        CONCUR   NO
        SUBDOC   NO
        FORMAL   YES
      APPINFO    "SDA"  -- conforming SGML Document Access
                           application --
    >
```

12.4 HTML Level 0 DTD

This is the Document Type Definition for the HyperText Markup Language as used by minimally conforming World Wide Web applications (HTML Level 0 DTD). Note that it is defined in terms of the Level 2 DTD.

```
<!-- html-0.dtd

     Document Type Definition for the HyperText Markup Language
     as used by minimally conforming World Wide Web applications
     (HTML Level 0 DTD).

-->

<!ENTITY % HTML.Version
     "-//IETF//DTD HTML 2.0 Level 0//EN"
     -- public identifier for "minimal conformance" version --

     -- Typical usage:

         <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Level 0//EN">
         <html>
         ...
         </html>
```

```
     --
     >

<!-- Feature Test Entities -->

<!ENTITY % HTML.Highlighting "IGNORE">
<!ENTITY % HTML.Forms "IGNORE">

<!ENTITY % head.extra " ">
<!ENTITY % linkExtraAttributes " ">

<!ENTITY % html PUBLIC "-//IETF//DTD HTML 2.0//EN">
%html;
```

12.5 HTML Level 1 DTD

This is the Document Type Definition for the HyperText Markup Language with Level 1 Extensions (HTML Level 1 DTD). Note that it is defined in terms of the Level 2 DTD.

```
<!-- html-1.dtd

     Document Type Definition for the HyperText Markup Language
     with Level 1 Extensions (HTML Level 1 DTD).

-->

<!ENTITY % HTML.Version
     "-//IETF//DTD HTML 2.0 Level 1//EN"

     -- Typical usage:

         <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML Level 1//EN">
         <html>
         ...
         </html>
     --
     >

<!-- Feature Test Entities -->
<!ENTITY % HTML.Forms "IGNORE">

<!ENTITY % html PUBLIC "-//IETF//DTD HTML 2.0//EN">
%html;
```

12.6 HTML Level 2 DTD

This is the Document Type Definition for the HyperText Markup Language (HTML DTD):

```
<!--   html.dtd

       Document Type Definition for the HyperText Markup Language
                 (HTML DTD)

-->
```

```
<!ENTITY % HTML.Version
     "-//IETF//DTD HTML 2.0//EN"

     -- Typical usage:

         <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
         <html>
         ...
         </html>
     --
     >

<!--============ Feature Test Entities ======================-->

<!ENTITY % HTML.Recommended "IGNORE"
     -- Certain features of the language are necessary for
        compatibility with widespread usage, but they may
        compromise the structural integrity of a document.
        This feature test entity enables a more prescriptive
        document type definition that eliminates
        those features.
     -->

<![ %HTML.Recommended [
     <!ENTITY % HTML.Deprecated "IGNORE">
]]>

<!ENTITY % HTML.Deprecated "INCLUDE"
     -- Certain features of the language are necessary for
        compatibility with earlier versions of the specification,
        but they tend to be used an implemented inconsistently,
        and their use is deprecated. This feature test entity
        enables a document type definition that eliminates
        these features.
     -->

<!ENTITY % HTML.Highlighting "INCLUDE"
     -- Use this feature test entity to validate that a
        document uses no highlighting tags, which may be
        ignored on minimal implementations.
     -->

<!ENTITY % HTML.Forms "INCLUDE"
     -- Use this feature test entity to validate that a document
        contains no forms, which may not be supported in minimal
        implementations
     -->

<!--============== Imported Names ============================-->

<!ENTITY % Content-Type "CDATA"
     -- meaning an internet media type
        (aka MIME content type, as per RFC1521)
     -->
```

```
<!ENTITY % HTTP-Method "NAME"
     -- as per HTTP specification, in progress
     -->

<!ENTITY % URI "CDATA"
     -- The term URI means a CDATA attribute
        whose value is a Uniform Resource Identifier,
        as defined by "Universal Resource Identifiers"
        by Tim Berners-Lee, aka RFC 1630

        Note that CDATA attributes are limited by the LITLEN
        capacity (1024 in the current version of html.decl),
        so that URIs in HTML have a bounded length.

     -->

<!--========== DTD "Macros" ======================-->

<!ENTITY % heading "H1|H2|H3|H4|H5|H6">

<!ENTITY % list " UL | OL | DIR | MENU " >

<!--======== Character mnemonic entities =================-->

<!ENTITY % ISOlat1 PUBLIC
  "ISO 8879-1986//ENTITIES Added Latin 1//EN//HTML">
%ISOlat1;

<!ENTITY amp CDATA "&#38;"     -- ampersand          -->
<!ENTITY gt CDATA "&#62;"      -- greater than       -->
<!ENTITY lt CDATA "&#60;"      -- less than          -->
<!ENTITY quot CDATA "&#34;"    -- double quote       -->

<!--======= SGML Document Access (SDA) Parameter Entities =====-->

<!-- HTML 2.0 contains SGML Document Access (SDA) fixed
     attributes in support of easy transformation to the
     International Committee for Accessible Document Design
     (ICADD) DTD "-//EC-USA-CDA/ICADD//DTD ICADD22//EN".

     ICADD applications are designed to support usable access to
     structured information by print-impaired individuals through
     Braille, large print and voice synthesis.  For more
     information on SDA & ICADD:
        - ISO 12083:1993, Annex A.8, Facilities for Braille,
          large print and computer voice
        - ICADD ListServ
          <ICADD%ASUACAD.BITNET@ARIZVM1.ccit.arizona.edu>
        - Usenet news group bit.listserv.easi
        - Recording for the Blind, +1 800 221 4792
-->

<!ENTITY % SDAFORM  "SDAFORM  CDATA  #FIXED"
          -- one to one mapping       -->
```

```
<!ENTITY % SDARULE  "SDARULE  CDATA  #FIXED"
          -- context-sensitive mapping -->
<!ENTITY % SDAPREF  "SDAPREF  CDATA  #FIXED"
          -- generated text prefix     -->
<!ENTITY % SDASUFF  "SDASUFF  CDATA  #FIXED"
          -- generated text suffix     -->
<!ENTITY % SDASUSP  "SDASUSP  NAME   #FIXED"
          -- suspend transform process -->

<!--========== Text Markup ====================-->

<![ %HTML.Highlighting [

<!ENTITY % font " TT | B | I ">

<!ENTITY % phrase
           "EM | STRONG | CODE | SAMP | KBD | VAR | CITE ">

<!ENTITY % text "#PCDATA | A | IMG | BR | %phrase | %font">

<!ELEMENT (%font;|%phrase) - - (%text)*>
<!ATTLIST ( TT | CODE | SAMP | KBD | VAR )
     %SDAFORM; "Lit"
     >
<!ATTLIST ( B | STRONG )
     %SDAFORM; "B"
     >
<!ATTLIST ( I | EM | CITE )
     %SDAFORM; "It"
     >

<!-- <TT>       Typewriter text                          -->
<!-- <B>        Bold text                                -->
<!-- <I>        Italic text                              -->

<!-- <EM>       Emphasized phrase                        -->
<!-- <STRONG>   Strong emphais                           -->
<!-- <CODE>     Source code phrase                       -->
<!-- <SAMP>     Sample text or characters                -->
<!-- <KBD>      Keyboard phrase, e.g. user input         -->
<!-- <VAR>      Variable phrase or substituable          -->
<!-- <CITE>     Name or title of cited work              -->

<!ENTITY % pre.content "#PCDATA | A | HR | BR | %font | %phrase">

]]>

<!ENTITY % text "#PCDATA | A | IMG | BR">

<!ELEMENT BR    - O EMPTY>
<!ATTLIST BR
     %SDAPREF; "&#RE;"
     >
<!-- <BR>       Line break      -->
```

```
<!--========== Link Markup ======================-->

<![ %HTML.Recommended [
     <!ENTITY % linkName "ID">
]]>

<!ENTITY % linkName "CDATA">

<!ENTITY % linkType "NAMES"
     -- a list of these will be specified at a later date -->

<!ENTITY % linkExtraAttributes
    "REL %linkType #IMPLIED
     REV %linkType #IMPLIED
     URN CDATA #IMPLIED
     TITLE CDATA #IMPLIED
     METHODS NAMES #IMPLIED
    ">

<![ %HTML.Recommended [
     <!ENTITY % A.content   "(%text)*"
     -- <H1><a name="xxx">Heading</a></H1>
             is preferred to
        <a name="xxx"><H1>Heading</H1></a>
     -->
]]>

<!ENTITY % A.content   "(%heading|%text)*">

<!ELEMENT A     - - %A.content -(A)>
<!ATTLIST A
     HREF %URI #IMPLIED
     NAME %linkName #IMPLIED
     %linkExtraAttributes;
     %SDAPREF; "<Anchor: #AttList>"
     >
<!-- <A>                Anchor; source/destination of link    -->
<!-- <A NAME="...">     Name of this anchor                   -->
<!-- <A HREF="...">     Address of link destination           -->
<!-- <A URN="...">      Permanent address of destination      -->
<!-- <A REL=...>        Relationship to destination           -->
<!-- <A REV=...>        Relationship of destination to this   -->
<!-- <A TITLE="...">    Title of destination (advisory)       -->
<!-- <A METHODS="...">  Operations on destination (advisory)  -->

<!--========== Images ===========================-->

<!ELEMENT IMG    - O EMPTY>
<!ATTLIST IMG
     SRC %URI;  #REQUIRED
     ALT CDATA #IMPLIED
     ALIGN (top|middle|bottom) #IMPLIED
     ISMAP (ISMAP) #IMPLIED
```

```
     %SDAPREF; "<Fig><?SDATrans Img: #AttList>#AttVal(Alt)</Fig>"
     >

<!-- <IMG>              Image; icon, glyph or illustration   -->
<!-- <IMG SRC="...">    Address of image object              -->
<!-- <IMG ALT="...">    Textual alternative                  -->
<!-- <IMG ALIGN=...>    Position relative to text            -->
<!-- <IMG ISMAP>        Each pixel can be a link             -->

<!--========== Paragraphs=========================-->

<!ELEMENT P     - O (%text)*>
<!ATTLIST P
     %SDAFORM; "Para"
     >

<!-- <P>        Paragraph       -->

<!--========== Headings, Titles, Sections ===============-->

<!ELEMENT HR    - O EMPTY>
<!ATTLIST HR
     %SDAPREF; "&#RE;&#RE;"
     >

<!-- <HR>       Horizontal rule -->

<!ELEMENT ( %heading )  - -  (%text;)*>
<!ATTLIST H1
     %SDAFORM; "H1"
     >
<!ATTLIST H2
     %SDAFORM; "H2"
     >
<!ATTLIST H3
     %SDAFORM; "H3"
     >
<!ATTLIST H4
     %SDAFORM; "H4"
     >
<!ATTLIST H5
     %SDAFORM; "H5"
     >
<!ATTLIST H6
     %SDAFORM; "H6"
     >

<!-- <H1>       Heading, level 1 -->
<!-- <H2>       Heading, level 2 -->
<!-- <H3>       Heading, level 3 -->
<!-- <H4>       Heading, level 4 -->
<!-- <H5>       Heading, level 5 -->
<!-- <H6>       Heading, level 6 -->
```

```
<!--========== Text Flows ======================-->

<![ %HTML.Forms [
     <!ENTITY % block.forms "BLOCKQUOTE | FORM | ISINDEX">
]]>

<!ENTITY % block.forms "BLOCKQUOTE">

<![ %HTML.Deprecated [
     <!ENTITY % preformatted "PRE | XMP | LISTING">
]]>

<!ENTITY % preformatted "PRE">

<!ENTITY % block "P | %list | DL
        | %preformatted
        | %block.forms">

<!ENTITY % flow "(%text|%block)*">

<!ENTITY % pre.content "#PCDATA | A | HR | BR">
<!ELEMENT PRE - - (%pre.content)*>
<!ATTLIST PRE
     WIDTH NUMBER #implied
     %SDAFORM; "Lit"
     >

<!-- <PRE>              Preformatted text               -->
<!-- <PRE WIDTH=...>    Maximum characters per line     -->

<![ %HTML.Deprecated [

<!ENTITY % literal "CDATA"
     -- historical, non-conforming parsing mode where
        the only markup signal is the end tag in full
     -->

<!ELEMENT (XMP|LISTING) - -  %literal>
<!ATTLIST XMP
     %SDAFORM; "Lit"
     %SDAPREF; "Example:&#RE;"
     >
<!ATTLIST LISTING
     %SDAFORM; "Lit"
     %SDAPREF; "Listing:&#RE;"
     >

<!-- <XMP>              Example section         -->
<!-- <LISTING>          Computer listing        -->

<!ELEMENT PLAINTEXT - O %literal>
<!-- <PLAINTEXT>        Plain text passage      -->

<!ATTLIST PLAINTEXT
```

```
     %SDAFORM; "Lit"
     >
]]>

<!--========== Lists ==================-->

<!ELEMENT DL    - -  (DT | DD)+>
<!ATTLIST DL
     COMPACT (COMPACT) #IMPLIED
     %SDAFORM; "List"
     %SDAPREF; "Definition List:"
     >

<!ELEMENT DT    - O (%text)*>
<!ATTLIST DT
     %SDAFORM; "Term"
     >

<!ELEMENT DD    - O %flow>
<!ATTLIST DD
     %SDAFORM; "LItem"
     >

<!-- <DL>               Definition list, or glossary    -->
<!-- <DL COMPACT>       Compact style list              -->
<!-- <DT>               Term in definition list         -->
<!-- <DD>               Definition of term              -->

<!ELEMENT (OL|UL) - -  (LI)+>
<!ATTLIST OL
     COMPACT (COMPACT) #IMPLIED
     %SDAFORM; "List"
     >
<!ATTLIST UL
     COMPACT (COMPACT) #IMPLIED
     %SDAFORM; "List"
     >
<!-- <UL>               Unordered list                  -->
<!-- <UL COMPACT>       Compact list style              -->
<!-- <OL>               Ordered, or numbered list       -->
<!-- <OL COMPACT>       Compact list style              -->


<!ELEMENT (DIR|MENU) - -  (LI)+ -(%block)>
<!ATTLIST DIR
     COMPACT (COMPACT) #IMPLIED
     %SDAFORM; "List"
     %SDAPREF; "<LHead>Directory</LHead>"
     >
<!ATTLIST MENU
     COMPACT (COMPACT) #IMPLIED
     %SDAFORM; "List"
     %SDAPREF; "<LHead>Menu</LHead>"
     >
```

```
<!-- <DIR>              Directory list                  -->
<!-- <DIR COMPACT>      Compact list style              -->
<!-- <MENU>             Menu list                       -->
<!-- <MENU COMPACT>     Compact list style              -->

<!ELEMENT LI    - O %flow>
<!ATTLIST LI
     %SDAFORM; "LItem"
     >

<!-- <LI>               List item                       -->

<!--========== Document Body ====================-->

<![ %HTML.Recommended [
     <!ENTITY % body.content "(%heading|%block|HR|ADDRESS|IMG)*"
     -- <h1>Heading</h1>
        <p>Text ...
             is preferred to
        <h1>Heading</h1>
        Text ...
     -->
]]>

<!ENTITY % body.content "(%heading | %text | %block |
                          HR | ADDRESS)*">

<!ELEMENT BODY O O  %body.content>

<!-- <BODY>     Document body   -->

<!ELEMENT BLOCKQUOTE - - %body.content>
<!ATTLIST BLOCKQUOTE
     %SDAFORM; "BQ"
     >

<!-- <BLOCKQUOTE>       Quoted passage  -->

<!ELEMENT ADDRESS - - (%text|P)*>
<!ATTLIST  ADDRESS
     %SDAFORM; "Lit"
     %SDAPREF; "Address:&#RE;"
     >

<!-- <ADDRESS>  Address, signature, or byline   -->

<!--======= Forms ====================-->

<![ %HTML.Forms [

<!ELEMENT FORM - - %body.content
                  -(FORM) +(INPUT|SELECT|TEXTAREA)>
```

```
<!ATTLIST FORM
     ACTION %URI #IMPLIED
     METHOD (%HTTP-Method) GET
     ENCTYPE %Content-Type; "application/x-www-form-urlencoded"
     %SDAPREF; "<Para>Form:</Para>"
     %SDASUFF; "<Para>Form End.</Para>"
     >

<!-- <FORM>                      Fill-out or data-entry form  -->
<!-- <FORM ACTION="...">         Address for completed form   -->
<!-- <FORM METHOD=...>           Method of submitting form    -->
<!-- <FORM ENCTYPE="...">        Representation of form data  -->

<!ENTITY % InputType "(TEXT | PASSWORD | CHECKBOX |
                       RADIO | SUBMIT | RESET |
                       IMAGE | HIDDEN )">
<!ELEMENT INPUT - O EMPTY>
<!ATTLIST INPUT
     TYPE %InputType TEXT
     NAME CDATA #IMPLIED
     VALUE CDATA #IMPLIED
     SRC %URI #IMPLIED
     CHECKED (CHECKED) #IMPLIED
     SIZE CDATA #IMPLIED
     MAXLENGTH NUMBER #IMPLIED
     ALIGN (top|middle|bottom) #IMPLIED
     %SDAPREF; "Input: "
     >

<!-- <INPUT>                  Form input datum               -->
<!-- <INPUT TYPE=...>         Type of input interaction      -->
<!-- <INPUT NAME=...>         Name of form datum             -->
<!-- <INPUT VALUE="...">      Default/initial/selected value -->
<!-- <INPUT SRC="...">        Address of image               -->
<!-- <INPUT CHECKED>          Initial state is "on"          -->
<!-- <INPUT SIZE=...>         Field size hint                -->
<!-- <INPUT MAXLENGTH=...>    Data length maximum            -->
<!-- <INPUT ALIGN=...>        Image alignment                -->

<!ELEMENT SELECT - - (OPTION+) -(INPUT|SELECT|TEXTAREA)>
<!ATTLIST SELECT
     NAME CDATA #REQUIRED
     SIZE NUMBER #IMPLIED
     MULTIPLE (MULTIPLE) #IMPLIED
     %SDAFORM; "List"
     %SDAPREF;
     "<LHead>Select #AttVal(Multiple)</LHead>"
     >

<!-- <SELECT>                   Selection of option(s)        -->
<!-- <SELECT NAME=...>          Name of form datum            -->
<!-- <SELECT SIZE=...>          Options displayed at a time   -->
<!-- <SELECT MULTIPLE>          Multiple selections allowed   -->
```

```
<!ELEMENT OPTION - O (#PCDATA)*>
<!ATTLIST OPTION
     SELECTED (SELECTED) #IMPLIED
     VALUE CDATA #IMPLIED
     %SDAFORM; "LItem"
     %SDAPREF;
     "Option: #AttVal(Value) #AttVal(Selected)"
     >

<!-- <OPTION>                A selection option              -->
<!-- <OPTION SELECTED>       Initial state                  -->
<!-- <OPTION VALUE="...">    Form datum value for this option-->

<!ELEMENT TEXTAREA - - (#PCDATA)* -(INPUT|SELECT|TEXTAREA)>
<!ATTLIST TEXTAREA
     NAME CDATA #REQUIRED
     ROWS NUMBER #REQUIRED
     COLS NUMBER #REQUIRED
     %SDAFORM; "Para"
     %SDAPREF; "Input Text -- #AttVal(Name): "
     >

<!-- <TEXTAREA>                An area for text input      -->
<!-- <TEXTAREA NAME=...>       Name of form datum          -->
<!-- <TEXTAREA ROWS=...>       Height of area              -->
<!-- <TEXTAREA COLS=...>       Width of area               -->

]]>

<!--======= Document Head ======================-->

<![ %HTML.Recommended [
        <!ENTITY % head.extra "META* & LINK*">
]]>

<!ENTITY % head.extra "NEXTID? & META* & LINK*">

<!ENTITY % head.content "TITLE & ISINDEX? & BASE? &
                         (%head.extra)">

<!ELEMENT HEAD O O  (%head.content)>

<!-- <HEAD>     Document head   -->

<!ELEMENT TITLE - -  (#PCDATA)*>
<!ATTLIST TITLE
     %SDAFORM; "Ti"    >

<!-- <TITLE>    Title of document -->

<!ELEMENT LINK - O EMPTY>
<!ATTLIST LINK
     HREF %URI #REQUIRED
     %linkExtraAttributes;
```

```
     %SDAPREF; "Linked to : #AttVal (TITLE) (URN) (HREF)>"
     >

<!-- <LINK>                Link from this document             -->
<!-- <LINK HREF="...">     Address of link destination         -->
<!-- <LINK URN="...">      Lasting name of destination         -->
<!-- <LINK REL=...>        Relationship to destination         -->
<!-- <LINK REV=...>        Relationship of destination to this -->
<!-- <LINK TITLE="...">    Title of destination (advisory)     -->
<!-- <LINK METHODS="..."> Operations allowed (advisory)       -->

<!ELEMENT ISINDEX - O EMPTY>
<!ATTLIST ISINDEX
     %SDAPREF;
     "<Para>[Document is indexed/searchable.]</Para>">

<!-- <ISINDEX>          Document is a searchable index       -->

<!ELEMENT BASE - O EMPTY>
<!ATTLIST BASE
     HREF %URI; #REQUIRED
     >

<!-- <BASE>             Base context document                -->
<!-- <BASE HREF="...">  Address for this document           -->

<!ELEMENT NEXTID - O EMPTY>
<!ATTLIST NEXTID
     N %linkName #REQUIRED
     >

<!-- <NEXTID>           Next ID to use for link name         -->
<!-- <NEXTID N=...>     Next ID to use for link name         -->

<!ELEMENT META - O EMPTY>
<!ATTLIST META
     HTTP-EQUIV  NAME    #IMPLIED
     NAME        NAME    #IMPLIED
     CONTENT     CDATA   #REQUIRED    >

<!-- <META>                     Generic Metainformation      -->
<!-- <META HTTP-EQUIV=...>      HTTP response header name    -->
<!-- <META NAME=...>            Metainformation name         -->
<!-- <META CONTENT="...">       Associated information       -->

<!--======= Document Structure ================-->

<![ %HTML.Deprecated [
     <!ENTITY % html.content "HEAD, BODY, PLAINTEXT?">
]]>
<!ENTITY % html.content "HEAD, BODY">

<!ELEMENT HTML O O  (%html.content)>
<!ENTITY % version.attr "VERSION CDATA #FIXED '%HTML.Version;'">
```

```
<!ATTLIST HTML
     %version.attr;
     %SDAFORM; "Book"
     >

<!-- <HTML>                          HTML Document    -->
```

13. Character Entity Sets

The following entity names are used in HTML, always prefixed by ampersand (&) and followed by a semicolon as shown. They represent particular graphic characters which have special meanings in places in the markup, or may not be part of the character set available to the writer.

13.1 Numeric and Special Graphic Entity Set

The following table lists each of the supported characters specified in the Numeric and Special Graphic entity set, along with its name, syntax for use, and description. This list is derived from ISO Standard 8879:1986//ENTITIES Numeric and Special Graphic//EN. However, HTML does not provide support for the entire entity set — only the entities listed below are supported.

GLYPH	NAME	SYNTAX	DESCRIPTION
<	lt	<	Less than sign
>	gt	>	Greater than sign
&	amp	&	Ampersand
"	quot	"	Double quote sign

13.2 ISO Latin 1 Character Entity Set

The following table lists each of the characters specified in the Added Latin 1 entity set, along with its name, syntax for use, and description. This list is derived from ISO Standard 8879:1986//ENTITIES Added Latin 1//EN. HTML supports the entire entity set.

NAME	SYNTAX	DESCRIPTION
Aacute	Á	Capital A, acute accent
Agrave	À	Capital A, grave accent
Acirc	Â	Capital A, circumflex accent
Atilde	Ã	Capital A, tilde
Aring	Å	Capital A, ring
Auml	Ä	Capital A, dieresis or umlaut mark
AElig	Æ	Capital AE dipthong (ligature)
Ccedil	Ç	Capital C, cedilla
Eacute	É	Capital E, acute accent
Egrave	È	Capital E, grave accent
Ecirc	Ê	Capital E, circumflex accent
Euml	Ë	Capital E, dieresis or umlaut mark
Iacute	Í	Capital I, acute accent
Igrave	Ì	Capital I, grave accent
Icirc	Î	Capital I, circumflex accent
Iuml	Ï	Capital I, dieresis or umlaut mark
ETH	Ð	Capital Eth, Icelandic
Ntilde	Ñ	Capital N, tilde

```
Oacute   &Oacute;  Capital O, acute accent
Ograve   &Ograve;  Capital O, grave accent
Ocirc    &Ocirc;   Capital O, circumflex accent
Otilde   &Otilde;  Capital O, tilde
Ouml     &Ouml;    Capital O, dieresis or umlaut mark
Oslash   &Oslash;  Capital O, slash
Uacute   &Uacute;  Capital U, acute accent
Ugrave   &Ugrave;  Capital U, grave accent
Ucirc    &Ucirc;   Capital U, circumflex accent
Uuml     &Uuml;    Capital U, dieresis or umlaut mark
Yacute   &Yacute;  Capital Y, acute accent

THORN    &THORN;   Capital THORN, Icelandic
szlig    &szlig;   Small sharp s, German (sz ligature)

aacute   &aacute;  Small a, acute accent
agrave   &agrave;  Small a, grave accent
acirc    &acirc;   Small a, circumflex accent
atilde   &atilde;  Small a, tilde
aring    &aring;   Small a, ring
auml     &auml;    Small a, dieresis or umlaut mark
aelig    &aelig;   Small ae dipthong (ligature)
ccedil   &ccedil;  Small c, cedilla
eacute   &eacute;  Small e, acute accent
egrave   &egrave;  Small e, grave accent
ecirc    &ecirc;   Small e, circumflex accent
euml     &euml;    Small e, dieresis or umlaut mark
iacute   &iacute;  Small i, acute accent
igrave   &igrave;  Small i, grave accent
icirc    &icirc;   Small i, circumflex accent
iuml     &iuml;    Small i, dieresis or umlaut mark
eth      &eth;     Small eth, Icelandic
ntilde   &ntilde;  Small n, tilde
oacute   &oacute;  Small o, acute accent
ograve   &ograve;  Small o, grave accent
ocirc    &ocirc;   Small o, circumflex accent
otilde   &otilde;  Small o, tilde
ouml     &ouml;    Small o, dieresis or umlaut mark
oslash   &oslash;  Small o, slash
uacute   &uacute;  Small u, acute accent
ugrave   &ugrave;  Small u, grave accent
ucirc    &ucirc;   Small u, circumflex accent
uuml     &uuml;    Small u, dieresis or umlaut mark
yacute   &yacute;  Small y, acute accent
thorn    &thorn;   Small thorn, Icelandic
yuml     &yuml;    Small y, dieresis or umlaut mark
```

13.3 Character Octet Entity Set

This list, sorted numerically, is derived from ISO-8859-1 8-bit single-byte coded graphic character set:

```
REFERENCE          DESCRIPTION
&#00; - &#08;      Unused
&#09;              Horizontal tab
&#10;              Line feed
&#11; - &#31;      Unused

&#32;              Space
&#33;              Exclamation mark
&#34;              Quotation mark
&#35;              Number sign
&#36;              Dollar sign
&#37;              Percent sign
&#38;              Ampersand
&#39;              Apostrophe
&#40;              Left parenthesis
&#41;              Right parenthesis
&#42;              Asterisk
&#43;              Plus sign
&#44;              Comma
&#45;              Hyphen
&#46;              Period (fullstop)
&#47;              Solidus (slash)

&#48; - &#57;      Digits 0-9

&#58;              Colon
&#59;              Semi-colon
&#60;              Less than
&#61;              Equals sign
&#62;              Greater than
&#63;              Question mark
&#64;              Commercial at

&#65; - &#90;      Letters A-Z

&#91;              Left square bracket
&#92;              Reverse solidus (backslash)
&#93;              Right square bracket
&#94;              Caret
&#95;              Horizontal bar (underscore)
&#96;              Acute accent

&#97; - &#122;     Letters a-z

&#123;             Left curly brace
&#124;             Vertical bar
&#125;             Right curly brace
&#126;             Tilde
```

```
&#127; -     Unused

&#161;            Inverted exclamation
&#162;            Cent sign
&#163;            Pound sterling
&#164;            General currency sign
&#165;            Yen sign
&#166;            Broken vertical bar
&#167;            Section sign
&#168;            Umlaut (dieresis)
&#169;            Copyright
&#170;            Feminine ordinal
&#171;            Left angle quote, guillemotleft
&#172;            Not sign
&#173;            Soft hyphen
&#174;            Registered trademark
&#175;            Macron accent
&#176;            Degree sign
&#177;            Plus or minus
&#178;            Superscript two
&#179;            Superscript three
&#180;            Acute accent
&#181;            Micro sign
&#182;            Paragraph sign
&#183;            Middle dot
&#184;            Cedilla
&#185;            Superscript one
&#186;            Masculine ordinal
&#187;            Right angle quote, guillemotright
&#188;            Fraction one-fourth
&#189;            Fraction one-half
&#190;            Fraction three-fourths
&#191;            Inverted question mark

&#192;            Capital A, grave accent
&#193;            Capital A, acute accent
&#194;            Capital A, circumflex accent
&#195;            Capital A, tilde
&#196;            Capital A, dieresis or umlaut mark
&#197;            Capital A, ring
&#198;            Capital AE dipthong (ligature)
&#199;            Capital C, cedilla
&#200;            Capital E, grave accent
&#201;            Capital E, acute accent
&#202;            Capital E, circumflex accent
&#203;            Capital E, dieresis or umlaut mark
&#204;            Capital I, grave accent
&#205;            Capital I, acute accent
&#206;            Capital I, circumflex accent
&#207;            Capital I, dieresis or umlaut mark
&#208;            Capital Eth, Icelandic
&#209;            Capital N, tilde
&#210;            Capital O, grave accent
&#211;            Capital O, acute accent
```

```
&#212;          Capital O, circumflex accent
&#213;          Capital O, tilde
&#214;          Capital O, dieresis or umlaut mark

&#215;          Multiply sign

&#216;          Capital O, slash
&#217;          Capital U, grave accent
&#218;          Capital U, acute accent
&#219;          Capital U, circumflex accent
&#220;          Capital U, dieresis or umlaut mark
&#221;          Capital Y, acute accent

&#222;          Capital THORN, Icelandic
&#223;          Small sharp s, German (sz ligature)

&#224;          Small a, grave accent
&#225;          Small a, acute accent
&#226;          Small a, circumflex accent
&#227;          Small a, tilde
&#228;          Small a, dieresis or umlaut mark
&#229;          Small a, ring
&#230;          Small ae dipthong (ligature)
&#231;          Small c, cedilla
&#232;          Small e, grave accent
&#233;          Small e, acute accent
&#234;          Small e, circumflex accent
&#235;          Small e, dieresis or umlaut mark
&#236;          Small i, grave accent
&#237;          Small i, acute accent
&#238;          Small i, circumflex accent
&#239;          Small i, dieresis or umlaut mark
&#240;          Small eth, Icelandic
&#241;          Small n, tilde
&#242;          Small o, grave accent
&#243;          Small o, acute accent
&#244;          Small o, circumflex accent
&#245;          Small o, tilde
&#246;          Small o, dieresis or umlaut mark

&#247;          Division sign

&#248;          Small o, slash
&#249;          Small u, grave accent
&#250;          Small u, acute accent
&#251;          Small u, circumflex accent
&#252;          Small u, dieresis or umlaut mark
&#253;          Small y, acute accent
&#254;          Small thorn, Icelandic
&#255;          Small y, dieresis or umlaut mark
```

14. Security Considerations

Anchors, embedded images, and all other elements which contain URIs as parameters may cause the URI to be dereferenced in response to user input. In this case, the security considerations of the URI specification apply.

Documents may be constructed whose visible contents mislead the reader to follow a link to unsuitable or offensive material.

15. References

[1] T. Berners-Lee. "Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web." RFC 1630, CERN, June 1994.

[2] T. Berners-Lee, L. Masinter, and M. McCahill. "Uniform Resource Locators (URL)." RFC 1738, CERN, Xerox PARC, University of Minnesota, October 1994.

[3] T. Berners-Lee, R. T. Fielding, and H. Frystyk Nielsen. "Hypertext Transfer Protocol — HTTP/1.0." Work in Progress (draft-ietf-http-v10-spec-00.ps), MIT, UC Irvine, CERN, March 1995.

[4] N. Borenstein and N. Freed. "MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies." RFC 1521, Bellcore, Innosoft, September 1993.

[5] R. T. Fielding. "Relative Uniform Resource Locators." Work in Progress (draft-ietf-uri-relative-url-06.txt), UC Irvine, March 1995.

[6] C. F. Goldfarb. "The SGML Handbook." Y. Rubinsky, Ed., Oxford University Press, 1990.

[7] J. Postel. "Media Type Registration Procedure." RFC 1590, USC/ISI, March 1994.

[8] J. Reynolds and J. Postel. "Assigned Numbers." STD 2, RFC 1700, USC/ISI, October 1994.

[9] SoftQuad. "The SGML Primer." 3rd ed., SoftQuad Inc., 1991.

[10] US-ASCII. Coded Character Set – 7-Bit American Standard Code for Information Interchange. Standard ANSI X3.4-1986, ANSI, 1986.

[11] ISO 8859. International Standard -- Information Processing -- 8-bit Single-Byte Coded Graphic Character Sets -- Part 1: Latin Alphabet No. 1, ISO 8859-1:1987. Part 2: Latin alphabet No. 2, ISO 8859-2, 1987. Part 3: Latin alphabet No. 3, ISO 8859-3, 1988. Part 4: Latin alphabet No. 4, ISO 8859-4, 1988. Part 5: Latin/Cyrillic alphabet, ISO 8859-5, 1988. Part 6: Latin/Arabic alphabet, ISO 8859-6, 1987. Part 7: Latin/Greek alphabet, ISO 8859-7, 1987. Part 8: Latin/Hebrew alphabet, ISO 8859-8, 1988. Part 9: Latin alphabet No. 5, ISO 8859-9, 1990.

[12] ISO 8879. Information Processing — Text and Office Systems — Standard Generalized Markup Language (SGML), 1986.

16. Acknowledgments

The HTML document type was designed by Tim Berners-Lee at CERN as part of the 1990 World Wide Web project. In 1992, Dan Connolly wrote the HTML Document Type Definition (DTD) and a brief HTML specification.

Since 1993, a wide variety of Internet participants have contributed to the evolution of HTML, which has included the addition of in-line images introduced by the NCSA Mosaic software for WWW. Dave Raggett played an important role in deriving the FORMS material from the HTML+ specification.

Dan Connolly and Karen Olson Muldrow rewrote the HTML Specification in 1994. The document was then edited by the HTML working group as a whole, with updates being made by Eric Schieler, Mike Knezovich, and Eric W. Sink at Spyglass, Inc. Finally, Roy Fielding restructured the entire draft into its current form.

Special thanks to the many people who have contributed to this specification:

Terry Allen	Marc Andreessen
Tim Berners-Lee	Paul Burchard
James Clark	Daniel W. Connolly
Roy T. Fielding	Peter Flynn
Jay Glicksman	Paul Grosso
Eduardo Gutentag	Bill Hefley
Chung-Jen Ho	Mike Knezovich
Tom Magliery	Murray Maloney
Larry Masinter	Karen Olson Muldrow
Bill Perry	Dave Raggett
E. Corprew Reed	Yuri Rubinsky
Eric Schieler	James L. Seidman
Eric W. Sink	Stuart Weibel
Chris Wilson	François Yergeau

17. Authors' Addresses

Tim Berners-Lee
Director, W3 Consortium
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, U.S.A.
Tel: +1 (617) 253 9670
Fax: +1 (617) 258 8682
Email: timbl@w3.org

Daniel W. Connolly
Research Technical Staff, W3 Consortium
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, U.S.A.
Fax: +1 (617) 258 8682
Email: connolly@w3.org
URL: http://www.w3.org/hypertext/WWW/People/Connolly/

Appendices

These appendices are provided for informational reasons only — they do not form a part of the HTML specification.

A. Obsolete Features

This section describes elements that are no longer part of HTML. Client implementors should implement these obsolete elements for compatibility with previous versions of the HTML specification.

A.1 Comment Element

The Comment element is used to delimit unneeded text and comments. The Comment element has been introduced in some HTML applications but should be replaced by the SGML comment feature in new HTML interpreters (see Section 2.2.5).

A.2 Highlighted Phrase Element

<HP>

The Highlighted Phrase element should be ignored if not implemented. This element has been replaced by more meaningful elements (see Section 8).

Example of use:

```
<HP1>first highlighted phrase</HP1>non-highlighted
text<HP2>second highlighted phrase</HP2> etc.
```

A.3 Plain Text Element

<PLAINTEXT>

The Plain Text element is used to terminates the HTML entity and to indicate that what follows is not SGML which does not require parsing. Instead, an old HTTP convention specified that what followed was an ASCII (MIME "text/plain") body. Its presence is an optimization. There is no closing tag.

Example of use:

```
<PLAINTEXT>
0001 This is line one of a long listing
0002 file from <ANY@HOST.INC.COM> which is sent
```

A.4 Example and Listing Elements

<XMP> ... </XMP> and <LISTING> ... </LISTING>

The Example and Listing elements have been replaced by the Preformatted Text element (Section 10.2).

These styles allow text of fixed-width characters to be embedded absolutely as is into the document. The syntax is:

```
<LISTING> ... </LISTING>
```

or

```
<XMP> ... </XMP>
```

The text between these tags is typically rendered in a monospaced font so that any formatting done by character spacing on successive lines will be maintained.

Between the opening and closing tags:

- The text may contain any ISO Latin-1 printable characters, expect for the end tag opener. The Example and Listing elements have historically used specifications which do not conform to SGML. Specifically, the text may contain ISO Latin printable characters, including the tag opener, as long it they does not contain the closing tag in full.
- SGML does not support this form. HTML interpreters may vary on how they interpret other tags within Example and Listing elements.
- Line boundaries within the text are rendered as a move to the beginning of the next line, except for one immediately following a start tag or immediately preceding an end tag.
- The horizontal tab character must be interpreted as the smallest positive nonzero number of spaces which will leave the number of characters so far on the line as a multiple of 8. Its use is not recommended.

The Listing element is rendered so that at least 132 characters fit on a line. The Example element is rendered to that at least 80 characters fit on a line but is otherwise identical to the Listing element.

B. Proposed Features

This section describes proposed HTML elements and entities that are not currently supported under HTML Levels 0, 1, or 2, but may be supported in the future.

B.1 Additional Character Entities

To indicate special characters, HTML uses entity or numeric representations. Additional character presentations are proposed:

CHARACTER	REPRESENTATION
Non-breaking space	
Soft-hyphen	
Registered	®
Copyright	©

B.2 Defining Instance Element

<DFN> ... </DFN>

The Defining Instance element indicates the defining instance of a term. The typical rendering is bold or bold italic. This element is not widely supported.

B.3 Strike Element

<STRIKE> ... </STRIKE>

The Strike element is proposed to indicate strikethrough, a font style in which a horizontal line appears through characters. This element is not widely supported.

B.4 Underline Element

<U> ... </U>

The Underline element is proposed to indicate that the text should be rendered as underlined. This proposed tag is not supported by all HTML interpreters.

Example of use:

```
The text <U>shown here</U> is rendered in the document as
underlined.
```