Work in Progress

Information Technology — HyperText Markup Language (ISO-HTML)
Technologies de l'information — Langage de balisage d'hypertexte

Roger Price* David Abrahamson†

March 23rd, 1997

Contents

*University of Massachusetts, Lowell
†Trinity College, Dublin

Warning: This Committee Draft has not yet been approved by the SC18 National Bodies.

1 Foreword

This Committee Draft was prepared by the working group WG8 of ISO/IEC JTC1/SC18. The New Work Item Proposal was approved at the SC18 level in November 1996.

SC18/WG8 works on this project in liaison with the Internet Engineering Task Force and the World Wide Web Consortium. The text of this Committee Draft follows closely the W3C's HTML 3.2 Recommendation specification edited by David Raggett `URL:http://www.w3.org/pub/WWW/TR`, and the W3C Working Draft dated 1996-04-22 describing work in progress on "Inserting objects into HTML" `URL:http://www.w3.org/pub/WWW/TR/WD-object.html`. The World Wide Web Consortium's Recommendation is based on specifications prepared by the Internet Engineering Task Force:

- IETF Proposed Standard RFC 1866 "HyperText Markup Language — 2.0" edited by Tim Berners-Lee and Daniel W. Connolly, September 22nd, 1995. `URL:ftp://ds.internic.net/rfc/rfc1866.txt`
- IETF RFC 1867 "Form-based File Upload in HTML" edited by E. Nebel and L. Masinter, November 1995. `URL:ftp://ds.internic.net/rfc/rfc1867.txt`
- IETF RFC 1942 "HTML Tables" edited by Dave Raggett, May 1996. `URL:ftp://ds.internic.net/rfc/rfc1942.txt`
- IETF Internet Draft "Internationalization of the HyperText Markup Language" edited by François Yergeau, Gavin T. Nicol, Glenn Adams and Martin J. Dürst, January, 1997. `URL:ftp://ds.internic.net/rfc/rfc2070.txt`

1.1 *Differences between the Committee Draft and W3C HTML 3.2*

This subclause is not a part of the Committee Draft. It is provided here to facilitate reading of this text, and will be removed.

This subclause provides a detailed list of the differences between the proposed ISO/IEC International Standard for the HyperText Markup Language - ISO-HTML and the W3C Recommendation for HTML 3.2.

1. The document element defined by the CD-proposal has the name "ISO-HTML" rather than "HTML" as defined by the W3C's HTML 3.2 Recommendation.

2. The CD-proposal extends the W3C's HTML 3.2 Recommendation to include the IETF HTML internationalization features defined by RFC 2070. The differences between this CD-proposal and the HTML internationalization specification are:

   (a) The internationalization specification provides the ICADD fixed attributes to support easy transformation to their DTD. Since these facilities are not included in W3C's HTML 3.2 Recommendation, they are not included in this CD-proposal. If required, they could be included through the "addon" interface.

   (b) The W3C's HTML 3.2 Recommendation includes the elements `<STRIKE>`, `<U>`, `<BIG>`, `<SMALL>`, `<SUB>` and `<SUP>` in the parameter entity %font;. These elements are not present in %font; in the internationalization specification. As a result `<SUB>` and `<SUP>` have a content model (%text;)+ rather than (#PCDATA) as called for by the internationalization specification.

   Note 1 *In this CD-proposal the parameter entity %font; has been replaced by %physical.styles;, and the parameter entity %phrase; has been replaced by %logical.styles;.*

   (c) The elements `<DFN>` and `<Q>` have been added to the parameter entity %logical.styles;. `<DFN>` was included in %phrase; by the W3C's HTML 3.2 Recommendation and `<Q>`, introduced by the internationalization specification, has been added to the parameter entity %logical.styles; in this proposal as a clarification of its role in the DTD.

   (d) The attribute value "`centre`" is permitted as an alternative to the value "`center`". There is no difference in meaning.

3. The CD-proposal conforms more strictly to the requirements of ISO 8879:1986 subclause 15.5 "Documentation Requirements".

4. The CD-proposal does not offer facilities classified as deprecated by W3C's HTML 3.2 Recommendation: `<XMP>`, `<LISTING>` and `<PLAINTEXT>` elements. The horizontal tab character, although deprecated, is automatically converted to a single space using the SHORTREF and USEMAP facilities.

5. The start tag and end tag omission allowed by W3C's HTML 3.2 Recommendation for the document element <HTML> and the major elements <HEAD> and <BODY> are not allowed by this CD-proposal.

6. The <DIR> and <MENU> elements have been omitted since in practice they are simply sugared syntax for the <UL> element.

7. The <CENTER> element has been omitted since it is a special case of the <DIV> element.

8. Parameter entity references in the DTD are systematically terminated by the refc ';'. See [Gol90, 9.4.5 p.352] for a discussion of the conditions for omission of the refc.

9. All uses of the ISO owner identifier for the SGML standard are now specified as "ISO 8879:1986". See [Gol90, 10.2.1.1 p.383].

10. The CD-proposal does not provide direct support for the <FONT> or <BASEFONT> elements. If such facilities are required, they could be defined by use of the

11. This CD-proposal defines a <NOP> element and a NOP attribute for use by the DTD (to ensure syntactically correct content models for unused interfaces). This element and attribute are not intended for use in documents.

12. The inline style attributes for the <BODY> and <HR> elements which were not provided by IETF RFC 1866 but which were added by W3C's HTML 3.2 Recommendation have been removed from this CD-proposal. If required, they could be included through the "addon" interfaces.

13. The use of the ';' character as the separator for name/value pairs in the form-urlencoded media type, encouraged but not mandated by IETF RFC 1866 is required by this CD-proposal.

14. The correct nesting of the sections identified by the <H1> through <H6> elements which was recommended by IETF RFC 1866 is required by this CD-proposal. This specification formalizes the notion of sections which have headers defined by the <H1> through <H6> elements and bodies defined by fully minimizable <B1> through <B6> elements. The correct nesting of sections is thus enforced by the DTD. The technique allows authors to omit the start and end tags of the section body elements, thus facilitating the use of historic authoring systems. The parameter entity %body.content; has been renamed %section.content; to better indicate its meaning.

15. In the <IMG> element, the ALT attribute, considered vital by W3C's HTML 3.2 Recommendation but optional, becomes mandatory.

16. In the <IMG> element the attributes WIDTH, HEIGHT, HSPACE, VSPACE and BORDER not present in IETF HTML 2.0 but introduced by W3C's HTML 3.2 Recommendation are omitted. If required, they could be included through the "addon" interface.

17. The ACCEPT attribute of the <INPUT> element, proposed by RFC 1867, is not included in this CD-proposal.

18. The inline style attributes for the <INPUT> element specified by IETF RFC 1866 are not included in this CD-proposal. If required, they could be included through the "addon" interface.

19. The "place holder" elements <SCRIPT> and <STYLE> are replaced by a head.extend interface technique which allows the addition of such facilities when they are defined.

20. The term "character set" used by W3C's HTML 3.2 Recommendation in the specification of the <TEXTAREA> element has been replaced by the term "coded character set" as defined by IETF RFC 1866.

21. The URN and METHODS attributes of the <A> element, included in IETF HTML 2.0 and the internationalization specification, but omitted by W3C's HTML 3.2 Recommendation have been omitted from this CD-proposal. If required, they could be included through the "addon" interface.

22. The NOSHADE, SIZE and WIDTH attributes of the <HR> element have been omitted. They may be added through use of a style sheet, or by use of the hr.addon interface.

23. The TYPE attribute defined by the W3C HTML 3.2 Recommendation for the <LI>, <OL> and <UL> elements to provide a suggested presentation style is not included in this CD-proposal. If required, they could be included through the "addon" interfaces for these elements.

24. The CD-proposal uses the W3C's <OBJECT> element specification in place of the <APPLET> element specification which appears in the W3C's HTML 3.2 Recommendation. The following facilities defined by the <OBJECT> element specification are not

included in this specification, but may be added using the "addon" interface:

(a) The extensions to the `<A>` element for shaped regions through the addition of `<SHAPE>`, `<COORDS>` and `<ISMAP>` elements.

(b) The SHAPES attribute used to indicate the use of shaped regions.

(c) The inline style attributes `WIDTH`, `HEIGHT`, `BORDER`, `HSPACE` and `VSPACE`.

25. A "justifying" element is an element carrying an `ALIGN` attribute which specifies horizontal placement relative to the current left and right margins. If nested, justifying elements give priority to the innermost object. Possible values are defined by the parameter entity

26. The `<CAPTION>`, `<IMG>`, `<OBJECT>` and `<TABLE>` elements also carry `ALIGN` attributes which define their position with respect to the current text baseline and margins. Each of these elements have different possible values for the `ALIGN` attribute. The `ALIGN` attribute of the `<TABLE>` element has the specified default value "`left`". This is called for by the text of HTML 3.2 but not enforced by the HTML 3.2 DTD.

27. %form.content; and %form.text; entities have been added, and the %form; entity has been renamed %form.fields;.

28. The following elements are not allowed to have empty content: `<ADDRESS>`, `<BDO>`, `<CAPTION>`, `<DT>`, `<H1>`...`<H6>`, `<MAP>`, `<OL>`, `<P>`, `<PRE>`, `<SELECT>`, `<SPAN>`, `<UL>`, %logical.styles; and %physical.styles;.

29. The fully structured DTD allows for the exact specification of a document's logical structure.

2 Introduction

The HyperText Markup Language is an SGML application conforming to International Standard ISO 8879 — Standard Generalized Markup Language. It provides a simple way of structuring hypertext documents which refer to one another and which collectively create an enormous "web" which continues to grow and evolve as many hypertext authors add and modify documents.

The web has expanded and browser developers have added additional features to the markup language such as new tags and new semantics for the tags. As a result, many documents have been created which can only be rendered faithfully on a limited number of browsers. Normal web practice is to hide any syntactic problems detected by the browsers and thus the reader is not always aware that the page being browsed is not faithful to the original authored document.

This Committee Draft has been developed in an effort to ensure that it will remain possible for an author to produce simple hypertext for the web and be confident that a conforming browser will be able to render the document faithfully. This specification represents a core of the language to be supported by all conforming browsers and provides techniques for extending the core that are SGML conformant and represent good SGML practice.

The language defined by this Committee Draft differs slightly from the W3C's HTML 3.2 Recommendation; specification. It omits all deprecated features of the language, and some other features whose role is purely cosmetic. This is done in preparation for the expected future introduction of style sheets. Certain optional facilities such as markup omission of the document element and the major elements have been removed to produce more robust texts in keeping with recognized good SGML practice. This does not reduce in any way the expressive power of the language. A minimal ISO-HTML document has the form:

```
<!DOCTYPE ISO-HTML PUBLIC
  "ISO 15xxx:199x
  //DTD HyperText Markup Language//EN"
[<!ENTITY % ISOhtme PUBLIC
  "ISO 15xxx//ENTITIES CD//EN" >
 <!ENTITY % ISOlat1 PUBLIC
  "ISO 8879:1986//ENTITIES Added Latin 1//EN">
 %ISOlat1;%ISOhtme;
]>
<ISO-HTML>
<HEAD>
<TITLE>Les unit\&eacute;s de base</TITLE>
```

...other head elements

```
</HEAD>
<BODY>
```

...document body

```
</BODY>
</ISO-HTML>
```

This Committee Draft follows the current World Wide Web convention of presenting element and attribute names in upper case, although there is no formal requirement for the practice.

In order to support world wide use of the markup language, the internationalization facilities specified by the IETF have been included in this Committee Draft in anticipation of their expected inclusion in the W3C specifications.

The elements of the ISO-HTML language are specified in this Committee Draft in alphabetical order but may also be reviewed in the following order which reflects more closely the structure of the DTD:

2.1 This document

This Committee Draft represents the editors' best efforts to capture the continuing development of the language. The editors' have copied text from

- the IETF RFC 1866 [BLC95],
- the W3C's HTML 3.2 Recommendation specification [Rag97],
- the W3C Working Draft dated 1996-04-22 describing work in progress on "Inserting objects into HTML" [KMS+97],
- the SGML Handbook [Gol90] where needed to ensure consistency with those specifications,
- the IETF HTML internationalisation specification [YNAD95].

2.1.1 Source markup

This document was prepared using ISO 8879 based technology. Details of the DTD used to structure this document may be found in "ISO/IEC TR 9573-11 Information processing — SGML support facilities — Techniques for using SGML — Part 11: Application at ISO Central Secretariat for International Standards and Reports".

3 Scope

The scope of this Committee Draft is a conforming application of SGML, ISO 8879:1986, which provides a simple and stable markup language for documents to be published on the World Wide Web. This Committee Draft also provides techniques whereby application, regional or industry-specific extensions may be added in a way which follows good SGML practice.

3.1 Specificity of the scope

Since it is expected that a wide range of products, user applications, recommendations and other standards may use this standard, the scope focuses on the general structuring aspects and provides only sufficient semantics to ensure that the structures are rendered in a familiar way.

The scope excludes any standardization of models, services, systems, protocols or applications which are likely to make use of the ISO-HTML language. This specification does not define the "look and feel" of any conforming product.

4 Conformance

4.1 Conforming ISO-HTML documents

A document which conforms to this Committee Draft shall

1. be a conforming SGML document,

   Note 2 Editors to NB experts: *This requirement is itself required by ISO 8879:1986 subclause 15.2.2 [Gol90, p.480].*

2. conform to the requirements of this Committee Draft.

4.2 Conforming ISO-HTML systems

A conforming ISO-HTML system is a conforming SGML system which is able to process any conforming ISO-HTML document.

Note 3 *This requires the system to be capable of processing any conforming ISO-HTML document that is not inconsistent with its "system declaration". In order to promote the maximum capability for document interchange among systems, every conforming system must be able to support the reference concrete syntax and the ISO-HTML capacity set [Gol90, p.481].*

4.2.1 Documentation of conforming ISO-HTML systems

Systems conforming to this Committee Draft shall display the ISO-HTML identification text prominently:

1. In a prominent location in the front matter of all publications (normally the title page and the cover page),
2. On all identifying display screens of programs,
3. In all promotional and training material.

and in the national language of the documentation.

The ISO-HTML system identification text is: "ISO-HTML, an SGML system conforming to International Standard ISO 8879 — Standard Generalised Markup Language."

Note 4 *These requirements are intended to help users apply knowledge gained on one SGML system to the use of other systems, not to inhibit friendly documentation and human-computer interfaces [Gol90, p.486].*

Note 5 Editors to NB experts: *The editors have deliberately omitted any reference to the national language to be used for the identification text. The issue is too complex, and not worth the trouble [Gol90, p.486 subclause 15.5.1 line 6].*

The documentation shall distinguish SGML constructs from ISO-HTML conventions and system functions, and shall identify the SGML constructs as being part of the Standard Generalized Markup Language.

Note 6 *The objective of this requirement is for the user to be aware of which constructs are common to all SGML systems, and which are unique to ISO-HTML. This will reduce the experienced user's learning time for a new system or application.*

Note 7 Editors to NB experts: *This requirement is itself required by ISO 8879:1986 subclause 15.5.2 [Gol90, p.486 line 23].*

The documentation shall cite ISO 8879:1986 as a reference for supported SGML constructs that are not specifically documented for the system.

For example, if, for simplicity's sake, only a subset of some function is presented in the documentation (such as omitting some of the options of the entity declaration), it shall be stated clearly that other options exist and can be found in the SGML International Standard [Gol90, p.487].

Note 8 Editors to NB experts: *Do we need to include text to cover the requirement of ISO 8879:1986 subclause 15.5.3 "Terminology"? [Gol90, p.487].*

4.2.2 Validating ISO-HTML systems

In addition to being a conforming ISO-HTML system, a system is also a validating ISO-HTML system if

1. It is a validating SGML parser as defined by ISO 8879:1986 subclause 15.4; and
2. It finds and reports an ISO-HTML error if one exits; and
3. Does not report an ISO-HTML error where none exists.

Note 9 Editors to NB experts: *The W3C's HTML 3.2 Recommendation specification does not say what the reportable HTML errors are.*

4.3 Character set conformance

Note 10 Editors to NB experts: *We have taken the same approach here as was taken by ISO 2022, and this subclause is based on ISO 2022 clause 3.*

The SGML declaration provided with this Committee Draft calls for the use of the Basic Multilingual Plane of the ISO/IEC 10646 Universal Multiple-Octet Coded Character Set (UCS). ISO/IEC 10646 addresses whole classes of provisions and it is not intended by this Committee Draft that they are all implemented in any user agent. As a result it is only practicable to envisage limited conformance to ISO/IEC 10646 as defined in this subclause.

Under limited conformance, the following is required:

1. When the characters described by ISO/IEC 10646 are used, they shall be implemented by the control functions, and with the meanings and coded representation specified in ISO/IEC 10646.
2. When two systems with different levels of implementation of the ISO/IEC 10646 coded character set are required to communicate with one another, they shall do so using using that part of the coded character set they have in common.

   Note 11 *This Committee Draft does not discuss the operation of the HTTP protocol [FGM+97].*

3. If a server is unable to express a document using the limited character set supported by the user agent, it should instead deliver a document in the limited character set explaining the impossibility.
4. Code positions that are either reserved for registration or reserved for future standardization shall not be used.
5. No registered escape sequence shall be used with a meaning different from that defined by ISO/IEC 10646.

The UTF-1 transformation format of ISO/IEC 10646, registered by IANA as ISO-10646-UTF-1, has been removed from the ISO/IEC 10646 and should not be used.

5 Normative references

To be completed.

6 Definitions

The definitions of ISO 8879:1986 apply to this specification.

Definition 1 ***Justifying element*** *An element which carries an `ALIGN` attribute used to specify horizontal justification of its content relative to the current left and right margins. The `ALIGN` attribute takes one of the following values:*

```
                 The set of characters
                 in this column forms
                 the character repertoire
    |-------------------->|
    | This function called|
    |"coded character set"|
    | by RFC 1866         |
,---|---------------------|---------------------------------,
| Code  |  Bit  |   Character      |ISOlat1 |Numeric  |Glyph|
|posit- |pattern|     name         |entity  |character|     |
|  ion  |(hexa) |                  |ref.    |reference|     |
|-------+-------+------------------+--------+---------+-----|
|    0  | 00    |Unused            |        | &#00;   |     |
|  ...  |       |                  |        |         |     |
|  121  | 79    |SMALL LETTER y    |        | &#121;  |  y  |
|  122  | 7A    |SMALL LETTER z    |        | &#122;  |  z  |
|  123  | 7B    |LEFT CURLY BRACKET|        | &#123;  |  {  |
|  ...  |       |                  |        |         |     |
|  199  | C7    |CAPITAL C, CEDILLA|&Ccedil;| &#199;  |     |
|  200  | C8    |CAPITAL E, GRAVE  |&Egrave;| &#200;  |     |
|  201  | C9    |CAPITAL E, ACUTE  |&Eacute;| &#201;  |     |
|  ...  |       |                  |        |         |     |
|  255  | FF    |SMALL y, UMLAUT   |&yuml;  | &#255;  |     |
'--|-------|-------------|----------------------------------'
   |       |             |
   |       |------------>|
   |       |  This function is called the "character encoding
   |       |  scheme" by RFC 1866.
   |       |<------------|
   |       |  This function is called "character set" by SGML
   |       |
   |       Called the "code set" by SGML
   Called the "character number" by SGML
```

Figure 1: Illustration of some character representation definitions. This illustration is based on the character set defined by ISO 8859-1:1987 "8-bit single-byte coded graphic character sets", Part 1: Latin alphabet No. 1.

- *LEFT: text is left justified, ragged right.*
- *CENTER: text is centered, ragged left and ragged right.*
- *CENTRE: a permitted alternative specification of CENTER.*
- *RIGHT: text is right justified, ragged left.*

The justifying elements specified by this Committee Draft are `<ADDRESS>`, `<BLOCKQUOTE>`, `<DIV>`, `<HR>`, `<H1>`, `<H2>`, `<H3>`, `<H4>`, `<H5>`, `<H6>`, `<LI>`, `<OL>`, `<P>`, `<TD>`, `<TH>`, `<TR>` and `<UL>`.

Definition 2 ***Boxing element*** *An element which carries an `ALIGN` attribute used to specify the position of a box around the visible rendered area. The boxing elements specified by this Committee Draft are `<IMG>`, `<OBJECT>` and `<TABLE>`.*

Definition 3 ***Browser*** *A User Agent whose main function is to present documents to the user.*

Definition 4 ***Character*** *(Source: RFC1866) An atom of information, for example a letter or a digit. Graphic characters have associated glyphs, whereas control characters have associated processing semantics. The multiple definitions and techniques for the representation of characters may be the source of confusion. Table 1 on page 11 shows some of the ideas involved.*

Definition 5 ***Character encoding scheme*** *(Source: RFC1866) A function whose domain is the set of sequences of octets, and whose range is the set* *of sequences of characters from a character repertoire; that is, a sequence of octets and a character encoding scheme determines a sequence of characters.*

Definition 6 ***Character repertoire*** *(Source: RFC1866) A finite set of characters; eg. the range of a coded character set.*

Definition 7 ***Code position*** *(Source: RFC1866) An integer; a coded character set and a code position from its domain determine a character.*

Definition 8 ***Coded character set*** *(Source: RFC1866) A function whose domain is a subset of the integers and whose range is a character repertoire. That is, for some set of integers (usually of the form 0, 1, 2, ..., N), a coded character set and an integer in that set determine a character. Conversely, a character and a coded character set determine the character's code position (or, in rare cases, a few code positions).*

Definition 9 ***CRLF*** *(Source: RFC1521) The sequence of the two ISO 646:1983 characters CR (13) and LF (10) which, taken together, in this order, denote a line break.*

Definition 10 ***Form data set*** *(Source: RFC 1866) A sequence of name/value pairs; the names are given by an ISO-HTML document and the values are given by the user.*

Definition 11 ***Fragment identifier*** *(Source: RFC1866) The portion of an `HREF` attribute following the '#' character which may modify the presentation of the destination of a hyperlink.*

Definition 12 ***ISO-HTML browser*** *Browser which presents ISO-HTML documents.*

Definition 13 ***ISO-HTML document*** *A document structured in accordance with this Committee Draft.*

Definition 14 ***Hyperlink*** *A relationship between two anchors, called the source and the target. The link goes from the source to the target. The source is also known as the tail, and the target is also known as the destination or head.*

7 Synonyms and abbreviations

The following symbols and abbreviations are used in this Committee Draft.

ISO-HTML Pertaining to this standard.

HTML 3.2 A specification for the Hypertext Markup Language developed by the World Wide Web Consortium.

HTTP IETF RFC 2068 HyperText Transfer Protocol.

HyTime One or more architectural forms as provided by [HyT92].

IANA Internet Assigned Numbers Authority. IANA is the central coordinator for the assignment of unique parameter values for Internet protocols. The IANA is chartered by the Internet Society (ISOC) and the Federal Network Council (FNC) to act as the clearinghouse to assign and coordinate the use of numerous Internet protocol parameters [RP94].

IETF Internet Engineering Task Force.

RFC Request for Comments. An Internet Engineering Task Force specification.

SGML Notation provided through use of [SGM86].

URI Universal Resource Identifier as defined by [BL94].

URL Uniform Resource Locator as defined by [BLMM94].

WWW World Wide Web

W3C World Wide Web Consortium, founded in 1994 to develop common standards for the evolution of the World Wide Web. It is an industry consortium, hosted by the Massachusetts Institute of Technology Laboratory for Computer Science (MIT/LCS) in the United States, the Institut National de Recherche en Informatique et en Automatique (INRIA) in Europe and the Keio University Shonan Fujisawa Campus in Asia `http://www.w3.org/`.

8 Requirements

This Committee Draft has been designed to satisfy the following requirements, as documented in ISO/IEC JTC1/SC18/WG8 N1852 dated 24 June 1996:

- Provide a minimum presentation architecture for SGML applications.
- Clarify relationships between ISO-HTML and SGML details, such as SGML declaration and minimization.

- Allow ISO-HTML to be used in environments where ISO standards are required.
- Allow ISO-HTML to be used as a base architecture for other SGML applications.
- Documents conforming to this Committee Draft should be viewable by browsers that conform to the W3C's HTML 3.2 Recommendation (or an appropriate later version of that specification).
- Define useful subsets of the W3C's HTML 3.2 Recommendation (or an appropriate later version of that specification).

Here ends the ISO/IEC contribution to this specification. The following clauses are edited versions of publicly available text copyrighted by the IETF and the W3C.

9 Internationalization

The early applications of HTML on the World-Wide Web were seriously restricted by their reliance on the ISO 8859-1:1987 coded character set, which is appropriate only for Western European languages. This Committee Draft provides facilities for a fuller internationalization, based on RFC 2070 [YNAD95]. The following subclauses specify the general internationalization features.

9.1 The ALIGN attribute when used for justification

The internationalization facilities use the `ALIGN` attribute to specify the horizontal alignment to be applied to content within the scope of the justifying elements: `<ADDRESS>`, `<BLOCKQUOTE>`, `<DIV>`, `<HR>`, `<H1>`, `<H2>`, `<H3>`, `<H4>`, `<H5>`, `<H6>`, `<LI>`, `<OL>`, `<P>`, `<TD>`, `<TH>`, `<TR>` and `<UL>`.

The `ALIGN` justification attribute takes the following values:

- `ALIGN=left`

  Render "flush left" or left justified. The right edge is ragged.

- `ALIGN=center`

  Render "centered". The left and right edges are ragged.

- `ALIGN=centre`

  A permitted alternative specification of ALIGN=center.

- `ALIGN=right`

  Render "flush right" or right justified. The left edge is ragged.

- `ALIGN=justify`

  Render fully justified on the left and right. There are no ragged edges.

In the case of nested justification specifications, the innermost containing specification shall apply. If a justifying `ALIGN` attribute is omitted, the value of the next outer justifying specification is used. If the `ALIGN` attribute is omitted from all containing justifying elements, the value "`left`" shall be used.

Note 12 *There are elements defined by this Committee Draft which have an ALIGN attribute for historical reasons, but are not justifying elements.*

9.1.1 Examples of justifying behaviour

1. In this example all the containing justification specifications are present, and the innermost applies.

```
<DIV ALIGN=left>
<OL ALIGN=justify>
<LI ALIGN=right>Important sentence ...
</OL>
</DIV>
```

   The "Important sentence" is right aligned.

2. In this example only some of the containing justification specifications are present, and the innermost specification applies.

```
<DIV ALIGN=left>
<OL>
<LI ALIGN=justify>
Important sentence ...
</OL>
</DIV>
```

   The "Important sentence" is fully justified.

3. In this example none of the containing justification specifications are present, hence the default value applies.

```
<DIV>
Important sentence ...
</DIV>
```

   The "Important sentence" is left justified.

9.2 The CLASS attribute

It is expected that many ISO-HTML documents will be created by automatic processes from SGML documents structured according to richer document type definitions, eg. the DTD for ISO documents specified in ISO TR 9573. Conversion to the ISO-HTML structure may lead to the loss of some information about document content. For example the notes in an ISO document are tagged using the `<NOTE>` element, rather than the `<P>` element. A rendering system might typically apply the ISO note style to the content of a `<NOTE>` element, making it distinct in the output presentation. There may be a need to retain this type of information during the conversion to ISO-HTML so that browsers with the capability may provide a distinctive rendering.

The `CLASS` attribute provides a means of attaching a sequence of SGML names to the element indicating the class to which the element belongs: typically the source element name if the document is the result of an automatic conversion from another document type.

Note 13 *This Committee Draft does not define values for the `CLASS` attribute.*

9.2.1 An example

A conversion process might convert `<NOTE>This text does not define values for the class attribute.</NOTE>` to `<P CLASS=note>This text does not define values for the class attribute.</P>`

9.3 The DIR attribute

The `DIR` attribute indicates the writing direction of text. With block-type elements the `DIR` attribute indicates the base writing direction of the text in the block. If the attribute is omitted, the writing directional is inherited from the parent element if present. With inline elements, it indicates the element start a new writing direction embedding level. Directional-embedding is used to handle nested changes in writing direction. A common need for embedding characters is to handle text that has been pasted from one bidirectional context to another.

9.3.1 An example

Here is an example of a case where direction-embedding is needed:

The following latin (shown in upper case) and arabic (shown in lower case) letters in backing store with the specified writing direction embeddings (LRE is shorthand for `<SPAN DIR=ltr>`, RLE for `<SPAN DIR=rtl>` and PDF for `</SPAN>`):

```
LRE A B RLE a b LRE C D PDF c d PDF E F PDF
```

result in the following rendering with square brackets showing the directional transitions:

```
[ A B [ d c [ C D ] b a ] E F ]
```

Note 14 *If the `<SPAN>` elements with `DIR` attributes were not used, this example would result in the following rendering, assuming an inherited direction of "`ltr`": `[ A B [ b a [ C D ] d c ] E F ]`. Notice that `b a` is on the left and `d c` is on the right unlike the example where writing direction embedding levels are used. Without writing direction embedding characters there would be at most two levels: a base directional level and a single counterflow level.*

9.4 The ID attribute

When documents with the same information content in different languages are created, there may be a need to identify elements in the documents as being the "same" element, differing only in the language employed.

The `ID` attribute provides a means of identifying the "same" element in different documents. Within each document, the elements' `ID` attribute is set to the same value.

9.5 The LANG attribute

The `LANG` attribute identifies a natural language spoken, sung, written or otherwise used by human beings for communication of information between people. Computer languages are explicitly excluded. The value of the `LANG` attribute is referred to as the "language tag".

The syntax and registry of language tags used by this Committee Draft is the same as that defined by RFC 1766 [Alv95], where a language tag is composed of one or more parts: a primary language tag and a possible empty series of subtags. For example, using RFC 822 EBNF:

```
language-tag = primary-tag *( "-" subtag )
primary-tag  = 1*8ALPHA
subtag       = 1*8ALPHA
```

Whitespace is not allowed within the tag and all tags are case insensitive. The namespace of language tags is administered by the IANA. Example tags include: `en`, `en-US`, `en-cockney`, `i-cherokee` and `x-pig-latin`.

Two-letter primary-tags are reserved for ISO 639 language abbreviations. This Committee Draft does not specify three-letter primary-tags, however their description may be found in the "Ethnologue" [Gri92]. Any two-letter initial sub-tag is an ISO 3166 country name.

In the context of this Committee Draft, a language tag is not to be interpreted as a single token, per RFC 1766, but as a hierarchy. For example, a user agent that adjusts its rendering according to language should recognize that it has a match when a language tag in a style sheet entry matches the initial portion of the language tag of an element. An exact match should be preferred. This interpretation allows an element marked up as, for instance, `en-US` to trigger styles corresponding to, in order of preference, US-english (`en-US`) or 'international' English (`en`).

Note 15 *Using the language tag as a hierarchy does not imply that all languages with a common prefix will be understood by those fluent in one or more of the languages; it simply allows the user to request commonality when desired.*

Note 16 *It is intended that any new element introduced in later versions or extensions to ISO-HTML will admit the `LANG` attribute, unless there is a compelling reason not to do so.*

The rendering of elements is meant to be controlled (in part) by the `LANG` attribute. Specific user preferences set within a browser should override the `LANG` attribute which in turn overrides the value specified by the `LANG` attribute of any enclosing element. If none of these are set, a suitable default, perhaps controlled by the user's locale, should be used to control rendering.

9.6 User agent unable to handle characters

If a user agent is unable to handle a character due, for example, to a lack of a resource such as a font, this should be indicated to the end user, clearly but unobtrusively.

Note 17 *Since some documents may contain many characters that cannot be rendered, showing an alert for each one is excessive.*

Note 18 *If a numeric representation of a missing character is given, its hexadecimal, not decimal, form is to be preferred since this is the form used in character set standards.*

9.7 Byte order

When an ISO-HTML text is transmitted directly in a multibyte representation, this Committee Draft recommends:

1. That it be transmitted in big-endian byte order — high order byte first.
2. That the document always begin with a ZERO-WIDTH NON-BREAKING SPACE character (hexadecimal FEFF) which, when byte-reversed becomes number FFFE, a character guaranteed never to be assigned. Thus a user agent receiving an FFFE as the first two octets of a text would know that bytes have to be reversed for the remainder of the text.

9.8 Use of named character references

It is often convenient for an author to use named character references to specify characters which are not available on the author's keyboard. Wherever named characters are used for characters specified by ISO 10646, the name used shall be that specified in ISO TR 9573.

Note 19 *ISO 9573 defines names for all the characters specified by ISO 10646. The intention of this requirement is to facilitate interchange.*

Example: `<EM>Liberté Égalité Fraternité!</EM>`

10 The DTD

The DTD provided by this Committee Draft has a public document type definition with the invocation:

```
<!ENTITY % ISO-HTML PUBLIC
  "ISO 15xxx:199x
   //DTD HyperText Markup Language//EN">
%ISO-HTML;
```

10.1 Alternate public document type definition

Many popular user agents do not support the use this Committee Draft makes of the document type declaration subset. To facilitate the use of ISO-HTML an alternate public document type definition is provided to specify a DTD which is totally self contained and contains all of the features of ISO-HTML without using the document type declaration subset.

```
<!ENTITY % ISO-HTML PUBLIC
  "ISO 15xxx:199x
   //DTD HyperText Markup Language Core//EN">
%ISO-HTML;
```

With the use of this public document type definition the minimum document becomes:

```
<!DOCTYPE ISO-HTML PUBLIC
  "ISO 15xxx:199x
  //DTD HyperText Markup Language Core//EN">
<ISO-HTML>
<HEAD>
<TITLE>Les unit&eacute;s de base</TITLE>
... <hp1>other head elements</hp1>
</HEAD>
<BODY>
... <hp1>document body</hp1>
</BODY>
</ISO-HTML>
```

11 Copyright protection

This document makes several references to industry and proprietary standards, products and publications. Such references are not normative, and do not imply endorsement by the ISO, IEC, or their national member bodies or affiliates. Any brand names or trademarks mentioned are the property of their respective owners.

The formal SGML definitions are part of the text of this Committee Draft and are protected by copyright held by the IETF and the W3C.

12 Hyperlinks

Note 20 Editors to NB experts: *This clause is based entirely on clause 7 in RFC 1866 HTML 2.0 [BLC95].*

Hyperlinks provide the mechanism which ties the web together and gives it an overall structure. They are the threads of gossamer glueing documents together. They have two ends, known as the *source anchor* and the *target anchor* which are both identified by their address: an absolute Uniform Resource Identifier (URI) [BL94], optionally followed by a '#' character and a further sequence of characters called a fragment identifier. For example:

```
http://www.xyz.org/Project.html
http://www.xyz.org/Project.html#Overview
```

Note 21 Editors to NB experts: *In RFC 1866, source anchors are called* tail *anchors, and target anchors are called* head *anchors as viewed from the current position in the hypertext document by a user agent. That is, prior to selecting a hypertext link its destination is seen as a target, but after selection and rendering the same position subsequently appears as a source.*

In an anchor address, the URL refers to a resource; it may be used in a variety of information retrieval protocols to obtain an entity that represents the resource, such as an ISO-HTML document. The fragment identifier, if present, marks some position in the resource.

Each of the following constructs provides the target anchor of a hyperlink or set of hyperlinks:

- The `<A>` element with the `HREF` attribute present.
- The `<IMG>` element.
- The `<LINK>` element.
- The `<ISINDEX>` element.
- The `<FORM>` element with `METHOD=get`.
- The `<INPUT>` element with the `SRC` attribute present.

These markup constructs refer to source anchors by URL, in either absolute or relative form, or to a fragment identifier, or to both.

In the case of a relative URL, the actual address referenced by the user agent is constructed by combining the specified relative address with an absolute base URL as in RFC 1808 [Fie95]. The base address is taken from the document's `<BASE>` element, if present; otherwise it is determined as in RFC 1808 [Fie95].

12.1 Accessing resources

Once the address specified by the target anchor is determined, the user agent may obtain a representation of the resource (now the source).

For example, if the base address of a document is `http://host/x/y.html` and the document contains:

```
<IMG SRC="../icons/abc.jpeg">
```

then the user agent uses the URL `http://host/icons/abc.jpeg` to access the resource.

Note 22 *The user agent constructs the destination URL by discarding the file identifier portion of the base address (if present).*

12.2 Activation of hyperlinks

An ISO-HTML user agent allows the user to navigate the content of a document and request activation of hyperlinks denoted by `<A>` elements. To activate a link, the user agent obtains a representation of the resource identified by the address specified in the source anchor. If the representation is another ISO-HTML document, navigation may begin again with this new document.

12.3 Simultaneous presentation of image resources

An ISO-HTML user agent may activate hyperlinks indicated by <IMG> elements concurrently with processing a document; that is, image hyperlinks may be processed without explicit request by the user. Image resources should be embedded in the presentation at the point of the source anchor, that is the <IMG> element. The same also holds for <INPUT> elements within forms.

12.4 Fragment identifiers

Any characters following a '#' character in a hypertext address constitute a fragment identifier. In particular, an address of the form `#fragment` refers to a target anchor in the document.

The meaning of fragment identifiers depends on the media type of the representation of the target anchor's resource. For `text/html` representations, it refers to the <A> element whose NAME attribute value is the same as the fragment identifier. Character matching is case sensitive. The target document shall have exactly one such element. The user agent should render the source anchor element by scrolling to and/or highlighting the text at the destination position.

For example, if the base URL is `http://host/x/` and the user activates the link denoted by the following markup:

```
See:
<A>HREF="app1.html\#bananas">Appendix 1</A>
for more details on bananas.
```

then the user agent accesses the resource identified by `http://host/x/app1.html` and assuming the resource is represented using the `text/html` media type, the user agent shall locate the <A> element whose NAME attribute has the value `bananas` and begin navigation at that point.

13 The extend and addon interfaces

This Committee Draft does not define facilities for specifying style in the body of a conforming document. However, authors of simple documents may wish to make use of such a facility and suppliers of user agents may wish to add the facility to the ISO-HTML document architecture. This is possible, and conveniently done using one of the *extend* or *addon* interfaces. Many extensions and addons may be defined in the future: this clause defines the general *extend* and *addon* interface mechanism and provides an example of its use.

There are three types of interfaces available to document designers:

- New element interface

  The `head.extend` and `body.extend` interfaces which allow the addition of new elements and their attributes.

- Additional attribute interface

  The *a.addon*, *address.addon*, *area.addon*, *base.addon*, *bdo.addon*, *big.addon*, *blockquote.addon*, *body.addon*, *bold.addon*, *br.addon*, *b1.addon*, *b2.addon*, *b3.addon*, *b4.addon*, *b5.addon*, *b6.addon*, *caption.addon*, *dd.addon*, *dfn.addon*, *div.addon*, *dl.addon*, *dt.addon*, *form.addon*, *head.addon*, *hr.addon*, *h1.addon*, *h2.addon*, *h3.addon*, *h4.addon*, *h5.addon*, *h6.addon*, *img.addon*, *input.addon*, *isindex.addon*, *iso-html.addon*, *italic.addon*, *li.addon*, *link.addon*, *literal.addon*, *map.addon*, *meta.addon*, *ol.addon*, *option.addon*, *p.addon*, *param.addon*, *pre.addon*, *select.addon*, *small.addon*, *span.addon*, *strike.addon*, *sub.addon*, *sup.addon*, *table.addon*, *textarea.addon*, *th.addon*, *title.addon*, *tr.addon*, *u.addon* and *ul.addon* interfaces which allow the addition of further attributes to existing elements.

- Notation interface

  The use of *notations* as defined by ISO 8879:1986.

Note 23 *This Committee Draft does not define any uses of the* extend *or* addon *interfaces, although an example is given to assist readers in their understanding of the SGML mechanisms used.*

Note 24 Editors to NB experts: *This mention of the use of SGML notations does not extend the proposed ISO-HTML standard. The full power of SGML is available to all conforming applications [Gol90, 15.2.2 p.480].*

13.1 Conformance of extensions

No *extend* or *addon* shall cause a document to be nonconforming if that document is conforming in the absence of the *extend* or *addon*.

Note 25 *This Committee Draft does not address the question of interdependencies between possibly incompatible extensions defined by other specifications.*

13.2 The new element interface

The `head.extend` and `body.extend` interfaces provide a means of extending the architecture of a document to include additional elements and attributes. The `head.extend` interface allows additions within the head of a document, at any point at which the `<META>` element may be used. The `body.extend` interface allows additions within the body of a document at any point where an `<A>` element might be used. The technique used is to extend the content models which contain the `<META>` and `<A>` elements.

By default the interfaces contain a `<NOP>` element which has no semantic value, but the interface may be redefined to include an additional DTD fragment in the document architecture.

The procedure for adding new elements is as follows:

1. Place the definitions of the new elements and their attributes in a DTD fragment and assign a public text identifier. The definitions in this DTD fragment may make use of the entities defined in the ISO-HTML entity set.

   Note 26 *The fragment may be in a single file, or in two or more files, for example one file for parameter entity definitions and another for element definitions.*

2. Place a new definition for the `head.extend` or `body.extend` parameter entity in the document type declaration subset. The definition should provide a content model for the new elements. Typically it is of the form `ELEMENT1 | ELEMENT2 ....` The definition will override the default definition `<NOP>` provided by the Committee Draft DTD entity set [Gol90, p.404 line 32]. More than one extension may be defined in the `head.extend` or `body.extend` parameter entities with declarations of the form `EXTEND1.EL1 | EXTEND1.EL2 | ... | EXTEND2.EL1 ....`

3. Place a parameter entity definition (or definitions) for the DTD fragment(s) containing definitions of the new elements and attributes in the document type declaration subset.

4. Invoke the `head.extend` or `body.extend` parameter entity.

5. Invoke the parameter entities for the DTD fragments at the end of the document type declaration subset.

13.3 The additional attribute interface

The *a.addon*, *address.addon*, *area.addon*, *base.addon*, *bdo.addon*, *big.addon*, *blockquote.addon*, *body.addon*, *bold.addon*, *br.addon*, *b1.addon*, *b2.addon*, *b3.addon*, *b4.addon*, *b5.addon*, *b6.addon*, *caption.addon*, *dd.addon*, *dfn.addon*, *div.addon*, *dl.addon*, *dt.addon*, *form.addon*, *head.addon*, *hr.addon*, *h1.addon*, *h2.addon*, *h3.addon*, *h4.addon*, *h5.addon*, *h6.addon*, *img.addon*, *input.addon*, *isindex.addon*, *iso-html.addon*, *italic.addon*, *li.addon*, *link.addon*, *literal.addon*, *map.addon*, *meta.addon*, *ol.addon*, *option.addon*, *p.addon*, *param.addon*, *pre.addon*, *select.addon*, *small.addon*, *span.addon*, *strike.addon*, *sub.addon*, *sup.addon*, *table.addon*, *textarea.addon*, *th.addon*, *title.addon*, *tr.addon*, *u.addon* and *ul.addon* interfaces provide a means of extending the set of attributes of an existing element. By default, the interfaces contain an empty string, ie. no additional attributes, but by redefining the interface, new attributes may be added to extend the document architecture.

The procedure for adding new attributes to an existing element X is as follows:

1. Place the definitions of the new attributes for the X element in an *X.addon* parameter entity in a DTD fragment and assign it a public text identifier. The definitions in this DTD fragment may make use of the entities defined in the ISO-HTML entity set.

   Note 27 *This DTD fragment may be shared with the one defined in subclause 13.2 on page 18.*

   This definition will override the default definition provided by the Committee Draft DTD entity set [Gol90, p.404 line 32].

2. Place a parameter entity definition for the DTD fragment in the document type declaration subset.

3. Invoke the parameter entities for the DTD fragments at the end of the document type declaration subset.

13.4 Use of the document type declaration subset

The extension techniques provided by this Committee Draft make use of the document type declaration subset which is a part of all SGML documents. However many historic World Wide Web browsers do not process the subset, so it is recommended that for wide public use, the designer of an extension consider repackaging the extension with the parameter entity set and the document type definition provided by this Committee Draft as a single entity (file) which can be referenced by a suitable public text identifier in the DOCTYPE declaration.

Such a repackaging will reduce the risk of error by users of the extension, and will allow the authority defining the package to provide better documentation and quality assurance, particularly when the package contains several extensions.

13.5 The NOP attribute

A small number of the elements defined by this specification have a single `NOP` attribute assigned to them.

This attribute is defined in the DTD to ensure that the DTD remains syntactically correct when the *addon* interface is not used. This attribute has no meaning and is not intended for use by authors or authoring systems.

14 Elements rendered as rectangles

The `<IMG>`, `<OBJECT>` and `<TABLE>` elements defined by this Committee Draft are usually rendered as rectangles with boxes around the visible area. Each of the elements which creates such a box is provided with an `ALIGN` attribute to specify how the rendered item is positioned with respect to the baseline of the text flow in which it occurs. A user agent may take into account only that text which has been placed on the line prior to the placement of the item in deciding the alignment of the enclosing box.

Each of the elements described in this clause use some of the attribute values from the following list:

- `ALIGN=top`

  Aligns the top of the rectangle with the top of the text rendered in the current line.

- `ALIGN=texttop`

  A permitted alternative specification for `ALIGN=top`.

- `ALIGN=middle`

  Aligns the middle of the rectangle with the current baseline.

- `ALIGN=textmiddle`

  Aligns the middle of the rectangle with a horizontal line drawn through the middle of the text rendered in the current line.

- `ALIGN=bottom`

  Aligns the bottom of the rectangle with the current baseline.

- `ALIGN=baseline`

  A permitted alternative specification for `ALIGN=bottom`.

- `ALIGN=textbottom`

  Aligns the bottom of the rectangle with a horizontal line drawn through the bottom of the text rendered in the current line.

- `ALIGN=left`

  Places the rectangle to the right of the current left margin, temporarily changing this margin. Subsequent text may be flowed along the rectangle's righthand side.

- `ALIGN=center`

  Places the rectangle after the end of the current line, centered between the left and right margins. Subsequent text starts at the beginning of the next line.

- `ALIGN=centre`

  A permitted alternative specification for `ALIGN=center`.

- `ALIGN=right`

  Places the rectangle to the left of the current right margin, temporarily changing this margin so that subsequent text is flowed along the rectangle's lefthand side.

Note 28 *The `<CAPTION>` element and the justifying elements in subsection 9.1 on page 13 also have an `ALIGN` attribute, but in these cases its meaning is different.*

15 The A element — Source and target anchors

15.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT A  - -  (%text;)* -(A) >
<!ATTLIST A
          CHARSET NAME    #IMPLIED
          CLASS   NAMES   #IMPLIED
          DIR     %dirn;  #IMPLIED
          HREF    %URL;   #IMPLIED
          ID      ID      #IMPLIED
```

```
LANG    NAME    #IMPLIED
NAME    CDATA   #IMPLIED
REL     CDATA   #IMPLIED
REV     CDATA   #IMPLIED
TITLE   CDATA   #IMPLIED
%a.addon; >
```

Note 29 Editors to NB experts: *The attributes* `REL` *and* `REV` *are declared by W3C's HTML 3.2 Recommendation to have type CDATA but IETF RFC 1866 declares them as NAMES. Which do we choose?*

15.2 Description

The `<A>` element defines hyperlink source and target anchors (see clause 12 on page 16). A source anchor requires the `HREF` attribute which provides the address of the target, while a target anchor requires the `NAME` attribute which identifies it as a location that may be used as a target in source anchors. At least one of the `HREF` and `NAME` attributes shall be present.

Good practice dictates that anchors should not be attached to markup, thus `<H1><A NAME="xxx">Heading</A></H1>` is preferred to `<A NAME="xxx"><H1>Heading</H1></a>`.

Note 30 *In the example given, the deprecated usage is also forbidden by the DTD, since it would not respect the required correct nesting of sections.*

Both start and end tags are required.

15.3 Attributes

The attributes of the `<A>` element are:

- CHARSET

  A hint to the user agent concerning the character encoding scheme used by the resource pointed to by the link. It should be the appropriate value of the MIME charset parameter for that resource [BF93].

- CLASS

  Please see 9.2 on page 14.

- DIR

  Please see 9.3 on page 14.

- HREF

  Gives the URL of the source anchor of a hyperlink. For example:

  ```
  The way to
  <A HREF="hands-on.html">happiness</A>.
  ```

- ID

  Please see 9.4 on page 14.

- LANG

  Please see 9.5 on page 14.

- NAME

  Gives the name of the anchor, and makes it available as a target for a hyperlink. The value shall be unique for the scope of the current document. For example:

  ```
  <H2><A NAME=mit>Hacker's Paradise</A></H2>
  ```

- REL

  The `REL` attribute gives the relationship(s) described by the hyperlink. The value is a whitespace separated list of relationship names. The semantics of link relationships are not specified in this document.

- REV

  Same as the `REL` attribute, but the semantics of the relationship 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.

- TITLE

  Suggests a title for the destination resource — advisory only. The `TITLE` attribute may be used:

  - for display prior to accessing the destination resource, for example, as a margin note or in a small box while the mouse is over the anchor, or while the document is being loaded;
  - for resources that do not include a title, such as graphics, plain text and Gopher menus, for use as a window title.

15.4 Example

```
...
Please see
<A NAME="intro" HREF="#details">here</A>
for further details, and then
<A HREF="http://www.acme.firm">go here</A>
to apply for the job.
...
<H3><A NAME="details">Job description</A></H3>
...
<P>
<A HREF="#intro">Return to introduction.</A>
```

16 The ADDRESS element — Author's address

16.1 Formal definition

```
<!ENTITY % justify "(left|center|centre
                    |right|justify)" >
<!ENTITY % dirn         "(ltr|rtl)" >
<!ELEMENT ADDRESS - - (%text;|P)+ >
<!ATTLIST ADDRESS
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %address.addon; >
```

16.2 Description

The <ADDRESS> element contains such information as address, signature and authorship, often at the beginning or end of the body of a document.

The <ADDRESS> element requires start and end tags. User agents should render the content with paragraph-breaks before and after. In the absence of a style-sheet, the <ADDRESS> element should be rendered in an italic typeface and may be indented.

16.3 Attributes

The <ADDRESS> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

16.4 Example

```
<ADDRESS>
Madame l'&eacute;ditrice en Chef<BR>
Le Clarion du Pays<BR>
1, rue Rotative<BR>
Ruritanie<BR>
T&eacute;l&eacute;phone: +800 12 34 56 789
</ADDRESS>
```

17 The AREA element — Region in image map

17.1 Formal definition

```
<!ENTITY % shape "(rect|circle|poly)" >
<!ELEMENT AREA  - O  EMPTY >
<!ATTLIST AREA
          ALT    CDATA    #REQUIRED
          COORDS CDATA     #IMPLIED
          HREF   %URL;     #IMPLIED
          NOHREF (nohref)  #IMPLIED
          SHAPE  %shape;       rect
          %area.addon; >
```

17.2 Description

The <AREA> element describes a region in an image map. It has no content.

The start tag is required, but the end tag shall be omitted.

17.3 Attributes

The <AREA> element has the following set of attributes:

- ALT

  Specifies a text label which may be displayed in a status line (when a pointing device is moved over the corresponding region in an image), or used for constructing a textual menu for non-graphical user agents. Authors are strongly recommended to provide meaningful ALT attributes to support interoperability with speech-based or text-only agents.

- COORDS

  Specifies the size and position of a region in an image. See the SHAPE attribute for further details.

- HREF

  Specifies the target hypertext link (behaviour) associated with a region.

- NOHREF

  Specifies that there is no target hypertext link (behaviour) associated with a region. This allows authors to create holes in regions.

- SHAPE

  Specifies the shape of a region and takes one of the following values:

- SHAPE=rect

  The region is rectangular and the `COORDS` attribute specifies the position of its top left and bottom right corners: `COORDS=`*`left-x,top-y,right-x,bottom-y`*, measured (in pixels) from the top left corner of the image.

- SHAPE=circle

  The region is circular and the `COORDS` attribute specifies the position of its centre and the length of its radius: `COORDS=`*`centre-x,centre-y,radius`*.

- SHAPE=poly

  The region is a polygon and the `COORDS` attribute specifies the position of its corners listed in clockwise order: `COORDS=`*`x1,y1, x2,y2, ...`*.

If an x or y value is given with a percent sign as suffix, the value will be interpreted as a percentage of the image's width or height, respectively. For example, `SHAPE=rect COORDS="0,0, 50%,100%"`.

One of `HREF` or `NOHREF` shall be specified.

17.4 Example

```
<AREA NOHREF
      SHAPE=circle
      COORDS="50\%,50\%,3"
      ALT="Bulls Eye">
```

See also the example of a `<MAP>` element in clause 57 on page 45.

18 The B element — Bold character style

18.1 Formal definition

```
<!ENTITY % physical.styles "B | ..." >
<!ENTITY % dirn          "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

18.2 Description

The contents of a `<B>` element should be rendered in a bold text style.

Both start and end tags are required.

18.3 Attributes

The `<B>` element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

18.4 Example

```
Thank you for <B>NOT SMOKING</B>
```

19 The BASE element — Base URL specification

19.1 Formal definition

```
<!ENTITY % URL "CDATA" >
<!ELEMENT BASE - O EMPTY >
<!ATTLIST BASE
          HREF %URL; #REQUIRED
          %base.addon; >
```

19.2 Description

The `<BASE>` element provides a base URL for dereferencing relative URLs, using the rules given by the URL specification [BLMM94].

In the absence of a `<BASE>` element the document URL should be used.

Note 31 *This is not necessarily the same as the URL used to request the document, as the base URL may be overridden by an HTTP header accompanying a document.*

Note 32 Editors to NB experts: *This clause appears to contain a normative reference to RFC 1738. How should this be handled?*

The start tag is required, but the end tag shall be omitted.

19.3 Attributes

The <BASE> element has the following required attribute:

- HREF

  The value is a Uniform Resource Locator as defined by IETF RFC 1808 and IETF RFC 1738.

19.4 Example

Given the <BASE> specification

```
<BASE HREF="http://www.acme.com/intro.html">
```

the image referenced by

```
<IMG SRC="icons/logo.jpeg">
```

is dereferenced to

```
http://www.acme.com/icons/logo.jpeg
```

20 The BDO element — Bidirectional override

20.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT BDO  - -  (%text;)+ >
<!ATTLIST BDO
          DIR  %dirn;   #REQUIRED
          LANG NAME      #IMPLIED
          %bdo.addon; >
```

20.2 Description

The <BDO> element contains unusual pieces of text in which directionality cannot be resolved from the context in an unambiguous fashion. For example in part numbers, formulas, telephone numbers, punctuation and other texts whose directionality cannot be determined by context alone.

Both start and end tags are required.

20.3 Attributes

The <BDO> element has the following attributes:

- DIR

  Please see subclause 9.3 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

20.4 Example

```
A palindrome appears the same whether rendered
left-to-write:
<BDO DIR=ltr>evil rats on no star live</BDO>
or right-to-left:
<BDO DIR=rtl>evil rats on no star live</BDO>.
```

21 The BIG element — Big character style

21.1 Formal definition

```
<!ENTITY % physical.styles "... | BIG | ..." >
<!ENTITY % dirn                    "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

21.2 Description

The contents of a <BIG> element should be rendered in a larger font than would otherwise be used.

Both start and end tags are required.

21.3 Attributes

The <BIG> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

21.4 Example

```
<BIG>What the large print giveth,</BIG>
<SMALL>the small print taketh away.</SMALL>
```

22 The BLOCKQUOTE element — Block quotation

22.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT BLOCKQUOTE - - %section.content; >
<!ATTLIST BLOCKQUOTE
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %blockquote.addon; >
```

22.2 Description

The <BLOCKQUOTE> element contains a quotation. A typical rendering might be with a slight left and right indent, and/or an italic font. The <BLOCKQUOTE> usually provides space above and below the quoted text.

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

Both start and end tags are required.

22.3 Attributes

The <BLOCKQUOTE> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

22.4 Example

This example quotes from article 129C of the European Union Treaty.

```
<BLOCKQUOTE>
Afin de r&eacute;aliser les objectifs
vis&eacute;s &agrave; l'article 129B,
la Communaut&eacute; :
<P>
met en oeuvre toute action qui peut
s'av&eacute;rer n&eacute;cessaire
pour assurer l'interoperabilit&eacute;
des r&eacute;seaux, en particulier dans le
domaine de l'harmonisation des normes
techniques ;
</BLOCKQUOTE>
Trait&eacute; sur l'Union Europ&eacute;enne,
Article 129 C.
```

23 The BODY element — Document body

23.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT BODY  - -
          (%section.content;, (H1,B1)* ) >
<!ATTLIST BODY
          CLASS NAMES    #IMPLIED
          DIR   %dirn;   #IMPLIED
          ID    ID       #IMPLIED
          LANG  NAME     #IMPLIED
          %body.addon; >
```

23.2 Description

The <BODY> element contains the textual flow of the document.

Both start and end tags are required.

23.3 Attributes

The <BODY> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

23.4 Example

See clause 53 on page 42 for an example of the use of the <BODY> element.

24 The BR element — Line break

24.1 Formal definition

```
<!ELEMENT BR  - O  EMPTY >
<!ATTLIST BR
          NOP  (nop)  #IMPLIED
          %br.addon; >
```

24.2 Description

The
 element specifies a line break between words. There is no content.

The start tag is required, but the end tag shall be omitted.

24.3 Attributes

This Committee Draft defines no attributes for the
 element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

24.4 Example

```
<ADDRESS>
ISO/IEC Copyright Office<BR>
P.O. Box 56<BR>
1211 Geneva 20<BR>
Switzerland
</ADDRESS>
```

25 The B1 element — Major section body

25.1 Formal definition

```
<!ELEMENT B1  O O
          (%section.content;, (H2,B2)* ) >
<!ATTLIST B1
          NOP  (nop)  #IMPLIED
          %b1.addon; >
```

25.2 Description

The <B1> element provides a formal definition for the body of a major section. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

25.3 Attributes

This Committee Draft defines no attributes for the <B1> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

26 The B2 element — Section body

26.1 Formal definition

```
<!ELEMENT B2  O O
          (%section.content;, (H3,B3)* ) >
<!ATTLIST B2
          NOP  (nop)  #IMPLIED
          %b2.addon; >
```

26.2 Description

The <B2> element provides a formal definition for the body of a section. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

26.3 Attributes

This Committee Draft defines no attributes for the <B2> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

27 The B3 element — Subsection body

27.1 Formal definition

```
<!ELEMENT B3  O O
          (%section.content;, (H4,B4)* ) >
<!ATTLIST B3
          NOP  (nop)  #IMPLIED
          %b3.addon; >
```

27.2 Description

The <B3> element provides a formal definition for the body of a subsection. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

27.3 Attributes

This Committee Draft defines no attributes for the <B3> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

28 The B4 element — Subsubsection body

28.1 Formal definition

```
<!ELEMENT B4  O O
          (%section.content;, (H5,B5)* ) >
<!ATTLIST B4
          NOP  (nop)  #IMPLIED
          %b4.addon; >
```

28.2 Description

The <B4> element provides a formal definition for the body of a subsubsection. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

28.3 Attributes

This Committee Draft defines no attributes for the <B4> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

29 The B5 element — Subsubsubsection body

29.1 Formal definition

```
<!ELEMENT B5  O O
          (%section.content;, (H6,B6)* ) >
<!ATTLIST B5
          NOP  (nop)  #IMPLIED
          %b5.addon; >
```

29.2 Description

The <B5> element provides a formal definition for the body of a subsubsubsection. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

29.3 Attributes

This Committee Draft defines no attributes for the <B5> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

30 The B6 element — Minor subsubsubsection body

30.1 Formal definition

```
<!ELEMENT B6  O O
          (%section.content;) >
<!ATTLIST B6
          NOP  (nop)  #IMPLIED
          %b6.addon; >
```

30.2 Description

The <B6> element provides a formal definition for the body of a minor subsubsubsection. This is a part of the mechanism by which the DTD specifies the required nesting of sections, and thus a validating system will detect nesting errors.

Since there are many historic HTML browsers which do not handle the section body elements, it is recommended that authors and authoring systems make use of the permitted omission of both start and end tags.

30.3 Attributes

This Committee Draft defines no attributes for the <B6> element that are available to the author of ISO-HTML documents, although the following attribute is present:

- NOP

  Please see subclause 13.5 on page 19.

31 The CAPTION element — Table caption

31.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT CAPTION - - (%text;)+ >
<!ATTLIST CAPTION
          ALIGN (top|bottom) #IMPLIED
          CLASS NAMES        #IMPLIED
          DIR   %dirn;       #IMPLIED
          ID    ID           #IMPLIED
          LANG  NAME         #IMPLIED
          %caption.addon; >
```

31.2 Description

The <CAPTION> element is used only in tables, where it contains a caption for a table.

Both start and end tags are required.

31.3 Attributes

The <CAPTION> element has the following attributes:

- ALIGN

  The ALIGN attribute has value top or bottom and indicates the preferred placement of the caption with respect to the table. If the ALIGN attribute is omitted, browsers usually place the caption at the top of the table.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

Note 33 *<CAPTION> is not a justifying element. Authors requiring horizontal text alignment should consider including a justifying element such as <P> within the content of the <CAPTION>.*

31.4 Example

See clause 75 on page 55 for an example of the use of the <CAPTION> element.

32 The CITE element — Citation

32.1 Formal definition

```
<!ENTITY % logical.styles "CITE | ..." >
<!ENTITY % dirn            "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

32.2 Description

The <CITE> element contains a citation or a reference to other sources.

Both start and end tags are required.

32.3 Attributes

The <CITE> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

32.4 Example

```
<CITE LANG="en-US">The buck stops here.</CITE>
```

33 The CODE element — Program code

33.1 Formal definition

```
<!ENTITY % logical.styles
           "... | CODE | ..." >
<!ENTITY % dirn   "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

33.2 Description

The <CODE> element contains extracts from program code.

Both start and end tags are required.

33.3 Attributes

The <CODE> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

  Note 34 *The LANG attribute is used for specifying human languages, not computer languages.*

33.4 Example

The list append function may be coded elegantly in Prolog as follows:

```
<CODE>append([],L,L) .</CODE>
<CODE>append([X|L1],L2,[X|L3]) :-
   append(L1,L2,L3) .</CODE>
```

34 The DD element — Definition data

34.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT DD  - O  %section.content; >
<!ATTLIST DD
          CLASS   NAMES    #IMPLIED
          DIR     %dirn;   #IMPLIED
          ID      ID       #IMPLIED
          LANG    NAME     #IMPLIED
          %dd.addon; >
```

34.2 Description

The <DD> element is used only in definition lists [clause 37 on page 29], and contains a definition for a term marked with a <DT> element. The definition is usually formatted as an indented paragraph after the term.

The start tag is required, but the end tag may be omitted.

34.3 Attributes

The <DD> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

34.4 Example

See clause 37 on page 29 for an example of the use of the <DD> element.

35 The DFN element — Defining instance

35.1 Formal definition

```
<!ENTITY % logical.styles
           "... | DFN | ..." >
<!ENTITY % dirn  "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

35.2 Description

The contents of the <DFN> element are the defining instance of a term.

Both start and end tags are required.

35.3 Attributes

The <DFN> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

35.4 Example

```
ICAO <DFN>International Civil Aviation
Organization</DFN> responsible ...
```

36 The DIV element — Document division

36.1 Formal definition

```
<!ELEMENT DIV  - -  %section.content; >
<!ATTLIST DIV
          ALIGN %justify; #IMPLIED
          %div.addon; >
```

36.2 Description

The <DIV> element may be used to structure documents into divisions. This Committee Draft makes no recommendation for the presentation of divisions.

Both start and end tags are required.

36.3 Attributes

The <DIV> element has the following attribute:

- ALIGN

  Please see subclause 9.1 on page 13.

36.4 Example

```
<BODY>
  ...
  <DIV>
    English Dictionary Terms
    <DL>
      ...
    </DL>
  </DIV>
  <DIV>
    Japanese Dictionary Terms
    <DL>
      ...
    </DL>
  </DIV>
  ...
</BODY>
```

37 The DL element — Definition list

37.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT DL  - -  (DT|DD)+ >
<!ELEMENT DT  - O  (%text;)+ >
<!ELEMENT DD  - O  %section.content; >
<!ATTLIST DL
          CLASS   NAMES     #IMPLIED
          COMPACT (compact) #IMPLIED
          DIR     %dirn;    #IMPLIED
          ID      ID        #IMPLIED
          LANG    NAME      #IMPLIED
          %dl.addon; >
```

37.2 Description

The <DL> element provides structure for a definition list: a list of terms and corresponding definitions.

The content is a non empty sequence of intermixed <DT> and <DD> elements, usually in pairs. Multiple <DT> may be paired with a single <DD> element and vice versa.

Definition lists are usually formatted with the term flushed to the margin and the definition, formatted paragraph style, indented to follow the term. If the <DT> term does not fit in the <DT> column (often one third of the display area), it may be extended across the page with the <DD> content moved onto the next line, or it may be wrapped onto successive lines in the <DT> column.

Both start and end tags are required.

37.3 Attributes

The <DL> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- COMPACT

  The optional `COMPACT` attribute suggests that a compact rendering be used, perhaps because the list items are small, or the entire list is large. Unless the `COMPACT` attribute is present, a user agent may leave white space between successive <DT>, <DD> groupings. The `COMPACT` attribute may also imply a reduction of the width of the <DT> column.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

37.4 Example

```
<DL compact>
<DT>Center
<DT>Centre
<DD>A point equidistant from all points
on the surface of a sphere
<DD>In some field sports the player who
holds the middle position on the field,
court, or forward line
</DL>
```

38 The DT element — Definition term

38.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT DT  - O  (%text;)+ >
<!ATTLIST DT
          CLASS   NAMES    #IMPLIED
          DIR     %dirn;   #IMPLIED
          ID      ID       #IMPLIED
          LANG    NAME     #IMPLIED
          %dt.addon; >
```

38.2 Description

The <DT> element is used only in definition lists [clause 37 on page 29] and contains a term to be defined. The term is typically formatted flush to the margin. If the <DT> term does not fit in the <DT> column (often one third of the display area), it may be extended across the page with the <DD> content moved onto the next line, or it may be wrapped onto successive lines in the <DT> column.

The start tag is required, but the end tag may be omitted.

38.3 Attributes

The <DT> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

38.4 Example

See clause 37 on page 29 for an example of the use of the <DT> element.

39 The EM element — Emphasized text

39.1 Formal definition

```
<!ENTITY % logical.styles "... | EM | ..." >
```

```
<!ENTITY % dirn                "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

39.2 Description

The contents of the <EM> element should be emphasized and are usually rendered in italics.

Both start and end tags are required.

39.3 Attributes

The <EM> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

39.4 Example

```
Trespassers <EM>will</EM> be prosecuted!
```

40 The FORM element — Forms

40.1 Formal definition

```
<!ENTITY % form.fields
           "INPUT | SELECT | TEXTAREA" >
<!ENTITY % form.text
           "%text; | %form.fields;" >
<!ENTITY % form.content
           "(%block; | %form.text;)+" >
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT FORM - - %form.content; -(FORM) >
<!ATTLIST FORM
          ACTION  %URL;      #REQUIRED
          CLASS   NAMES       #IMPLIED
          DIR     %dirn;      #IMPLIED
          ENCTYPE %Content-Type;
  "application/x-www-form-urlencoded"
          ID      ID          #IMPLIED
          LANG    NAME        #IMPLIED
          METHOD  %HTTP-Method;   get
          %form.addon; >
```

40.2 Description

The <FORM> element provides a template for a form data set and an associated method and action URL. A form data set is a sequence of name/value pair fields. The names are specified in the NAME attributes of form input elements, and the values are provided with initial values by various forms of markup, and may subsequently be edited by the user. The resulting form data set is used to access an information service as defined by the ACTION and METHOD attributes.

Form elements can be mixed in with document structuring elements. For example, a <FORM> element may contain lists which contain <INPUT> elements. This gives considerable flexibility in designing the layout of forms. <FORM> elements do not contain other <FORM> elements.

Usually <FORM> elements contain a sequence of <INPUT> elements along with additional document structure.

Both start and end tags are required.

40.2.1 Form submission

A user agent usually begins processing by presenting a document with the fields in their initial state. The user is allowed to modify the content of fields, constrained by the field type. When the user indicates that the form should be submitted (using for example a submit button), the form data set is processed as defined by its METHOD, ACTION URL and ENCTYPE attributes.

Where there is only one single-line text input field in a form, the user agent should accept "Enter" in that field as a request to submit the form.

40.3 Attributes

The attributes of the <FORM> element are:

- ACTION

  Specifies the action URL for the form. The action URL of a form defaults to the base URL of the document 12 on page 16.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ENCTYPE

  Specifies the media type used to encode the name/value pairs for transport, in case the protocol itself does not impose a format. The default encoding for all forms is `application/x-www-form-urlencoded` A form data set is represented in this media type as follows:

  1. The form field names and values are escaped; spaces are replaced by '+', and then reserved characters are escaped as per RFC 1738 [BLMM94]; that is non-alphanumeric characters are replaced by `%HH`, a percent sign and two hexadecimal digits representing the ASCII code of the character. Line breaks, as in multi-line text field values, are represented as CRLF pairs, i.e. `%0D%0A`.

     Note 35 Editors to NB experts: *How do we handle the reference to "ASCII"?*

  2. The fields are listed in the order they appear in the document with the name separated from the value by '=' and the pairs separated from each other by ';'. Fields with null values may be omitted. In particular, unselected radio buttons and checkboxes should not appear in the encoded data, but hidden fields with `VALUE` attributes present should.

     Note 36 Editors to NB experts: *RFC1866 allows the use of '&' instead of ';', but notes that this leads to ambiguities, since '&' is an SGML entity reference delimiter, and encourages the use of ';'.*

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- METHOD

  Selects a method for accessing the action URL. The set of applicable methods is a function of the scheme of the action URL specified for the form 40.2.1 on page 31. The `METHOD` attribute has one of the following values:

  – `METHOD=get` — Forms with no server side effects

    If the processing of a form is idempotent (ie. it has no lasting observable effect on the state of the world), then the form method should be `get`. Many database searches have no visible side-effects and make ideal applications for query forms.

    To process a form whose action URL is an HTTP URL and whose method is `get`, the user agent starts with the action URL and appends a '`?`' and the form data set, in `application/x-www-form-urlencoded` format. The user agent then traverses the link to this URL as if it were an anchor.

    Note 37 *The URL encoding may result in very long identifiers, which cause some historical HTTP server implementations to exhibit defective behaviour. As a result, some ISO-HTML forms may be written using `METHOD=post` even though the form submission has no server side effects.*

  – `METHOD=post` — Forms with server side effects

    If the service associated with the processing of a form has side effects (for example modification of a data base or subscription to a service), the method should be '`post`'.

    To process a form whose action URL is an HTTP URL and whose method is `post`, the user agent conducts an HTTP 'post' transaction using the action URL, and a message body of type `application/x-www-form-urlencoded` format. The user agent should display the response from the HTTP '`post`' interaction just as it would display the response from an HTTP '`get`'.

40.4 Example

```
<H1>Acme Company Inc</H1>
<H2>Smallsville Agency</H2>
<FORM METHOD=post
      ACTION="http://www.acme.firm/job">
<B>Job Application Form</B>
<BR>
Your Name:
<INPUT TYPE=text NAME=name
                 VALUE="Yosemite Sam">
<BR>
Do you have a valid driver licence
<BR>
<INPUT TYPE=radio NAME=driver
                  VALUE=yes>Yes
<INPUT TYPE=radio NAME=driver
```

```
                  VALUE=no CHECKED>No
<BR>
When you are finished,
you may submit this application:
<INPUT TYPE=submit VALUE="Apply">
<BR>
You may clear the application form
and start over at any time:
<INPUT TYPE=reset VALUE="New Form">
<BR>
<EM>Acme is an equal opportunity employer<EM>
</FORM>
```

41 The HEAD element — Document header

41.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT HEAD  - -
   (TITLE & ISINDEX? & BASE?)
  +(LINK | META | %head.extend;) >
<!ATTLIST HEAD
          CLASS NAMES   #IMPLIED
          DIR   %dirn;  #IMPLIED
          ID    ID      #IMPLIED
          LANG  NAME    #IMPLIED
          %head.addon; >
```

41.2 Description

The <HEAD> element defines the head of an ISO-HTML document, and contains an unordered collection of general information about the document.

Both start and end tags are required.

The content model of the <HEAD> element also provides the `head.extend` interface. see subclause 13.2 on page 18.

41.3 Attributes

The <HEAD> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

41.4 Example

See clause 53 on page 42 for an example of the use of the <HEAD> element.

42 The HR element — Horizontal rule

42.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT HR  - O  EMPTY >
<!ATTLIST HR
          ALIGN %justify; #IMPLIED
          DIR   %dirn;    #IMPLIED
          %hr.addon; >
```

42.2 Description

The <HR> element specifies a horizontal separator in a document. Visual user agents usually present it as a full width horizontal rule, while speech-based user agents might present it as a pause.

The start tag is required, but the end tag shall be omitted.

42.3 Attributes

The <HR> element has the following attributes:

- ALIGN

  Please see subclause 9.1 on page 13.

- DIR

  Please see subclause 9.3 on page 14.

42.4 Example

```
...
<H5>Monday Activities</H5>

    ...
  <HR>
<H5>Tuesday Activities</H5>

    ...
  <HR>
...
```

43 The H1 element — Major section header

43.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H1  - -  (%text;)* >
<!ATTLIST H1
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h1.addon; >
```

43.2 Description

The <H1> element specifies the beginning of a major section of a document and contains the title of that major section.

Both start and end tags are required.

This Committee Draft requires the correct nesting of sections. The <H1> element shall not be followed by an <H3>, <H4>, <H5>, or <H6> element without an intervening <H2> element.

43.3 Attributes

The <H1> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

43.4 Example

```
<H1>Continent</H1>
  ...
  <H2>Country</H2>
    ...
    <H3>Province</H3>
      ...
      <H4>County</H5>
        ...
        <H5>City</H4>
          ...
  <H3>State</H3>
    ...
    <H4>City</H4>
      ...
```

44 The H2 element — Section header

44.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H2  - -  (%text;)* >
<!ATTLIST H2
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h2.addon; >
```

44.2 Description

The <H2> element specifies the beginning of a section of a document and contains the title of the section.

Both start and end tags are required.

This Committee Draft requires correct nesting of sections. The <H2> element shall not be followed by an <H4>, <H5>, or <H6> element without an intervening <H3> element. An <H2> element shall be preceded by an <H1> element.

44.3 Attributes

The <H2> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

44.4 Example

See clause 43 on page 34 for an example of the use of header elements.

45 The H3 element — Subsection heade

45.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H3  - -  (%text;)* >
<!ATTLIST H3
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h3.addon; >
```

45.2 Description

The <H3> element specifies the beginning of a subsection of a document and contains the title of the subsection.

Both start and end tags are required.

This Committee Draft requires correct nesting of sections. The <H3> element shall not be followed by an <H5> or <H6> element without an intervening <H4> element. An <H3> element shall be preceded by an <H1> element.

45.3 Attributes

The <H3> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

45.4 Example

See clause 43 on page 34 for an example of the use of header elements.

46 The H4 element — Subsubsection header

46.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H4  - -  (%text;)* >
<!ATTLIST H4
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h4.addon; >
```

46.2 Description

The <H4> element specifies the beginning of a subsubsection of a document and contains the title of the subsubsection.

Both start and end tags are required.

This Committee Draft requires correct nesting of sections. The <H4> element shall not be followed by an <H6> element without an intervening <H5> element. An <H4> element shall be preceded by an <H1> element.

46.3 Attributes

The <H4> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

46.4 Example

See clause 43 on page 34 for an example of header elements.

47 The H5 element — Subsubsubsection header

47.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H5  - -  (%text;)* >
<!ATTLIST H5
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h5.addon; >
```

47.2 Description

The <H5> element specifies the beginning of a subsubsubsection of a document and contains the title of the subsubsubsection.

Both start and end tags are required.

This Committee Draft requires correct nesting of sections. An <H5> element shall be preceded by an <H1> element.

47.3 Attributes

The <H5> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

47.4 Example

See clause 43 on page 34 for an example of header elements.

48 The H6 element — Minor subsubsubsection header

48.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT H6  - -  (%text;)* >
<!ATTLIST H6
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %h6.addon; >
```

48.2 Description

The <H6> element specifies the beginning of a minor subsubsubsection of a document and contains the title of the minor subsubsubsection.

Both start and end tags are required.

This Committee Draft requires correct nesting of sections. An <H6> element shall be preceded by an <H1> element.

48.3 Attributes

The <H6> element has the following attributes.

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

48.4 Example

See clause 43 on page 34 for an example of header elements.

49 The I element — Italic character style

49.1 Formal definition

```
<!ENTITY % physical.styles "... | I | ..." >
<!ENTITY % dirn              "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

49.2 Description

The contents of an <I> element should be rendered in an italic text style if available, otherwise an alternative representation may be used.

Both start and end tags are required.

49.3 Attributes

The <I> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

49.4 Example

```
Following her <I LANG=fr>affaire</I> with the
well known rugby player, ...
```

50 The IMG element — Inline images

50.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT IMG  - O  EMPTY >
<!ATTLIST IMG
          ALIGN  (top|middle|bottom|left|right)
                                #IMPLIED
          ALT    CDATA   #REQUIRED
          CLASS  NAMES    #IMPLIED
          DIR    %dirn;   #IMPLIED
          ID     ID       #IMPLIED
          ISMAP  (ismap)  #IMPLIED
          LANG   NAME     #IMPLIED
          SRC    %URL;   #REQUIRED
          USEMAP %URL;    #IMPLIED
          %img.addon; >
```

50.2 Description

The <IMG> element refers to an image or icon via a hyperlink, see clause 12 on page 16.

Non-graphical user agents should process the value of the ALT attribute as an alternative to processing the image resource indicated by the SRC attribute.

The <IMG> element has no content.

The start tag is required, but the end tag shall be omitted.

50.3 Attributes

The <IMG> attributes are as follows:

- ALIGN

  Please see subclause 14 on page 19

- ALT

  The ALT attribute is required, and specifies text that may be used in place of the referenced image resource. Use of this attribute is vital for interoperability with speech-based and text only user agents.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ISMAP

  If the ISMAP attribute is present in an <IMG> element, the <IMG> element shall be contained in an <A> element with an HREF attribute present. This construct represents a set of hyperlinks. The user can choose from the set by selecting a pixel on the image. The user agent computes the target URL by appending the character '?' and the comma separated x and y coordinates of the selected pixel to the URL given in the <A> element.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- SRC

  The `SRC` attribute is required, and specifies the URL of the image resource.

  Note 38 *In practice, the media types of the image resources are limited to a few raster graphic formats: typically `image/jpeg`, `image/gif`. In particular, `text/html` resources are not intended to be used for images.*

- USEMAP

  Specifies a URL fragment identifier for a client-side image map defined with the `<MAP>` element, see subclause 57 on page 45.

50.4 Example

If the document contains:

```
...
<ISO-HTML><HEAD><TITLE>ImageMap Example</TITLE>
<BASE HREF="http://host/index"></HEAD>
<BODY>
Select a point of interest
in the following image:
<A HREF="/cgi-bin/imagemap">
   <IMG ismap SRC="atlas.jpeg"
        ALT="Atlas of central Europe">
</A>
...
```

and the user selects the upper-leftmost pixel in the image, the chosen hyperlink has the URL `http://host/cgi-bin/imagemap?0,0`

51 The INPUT element — User input field

51.1 Formal definition

```
<!ENTITY % dirn     "(ltr|rtl)" >
<!ENTITY % InputType "(checkbox|file
          |hidden|image|password|radio
          |reset|submit|text)" >
<!ELEMENT INPUT  - O  EMPTY >
<!ATTLIST INPUT
          CHECKED   (checked)   #IMPLIED
          CLASS     NAMES       #IMPLIED
          DIR       %dirn;      #IMPLIED
          ID        ID          #IMPLIED
          LANG      NAME        #IMPLIED
          MAXLENGTH NUMBER      #IMPLIED
          NAME      CDATA       #IMPLIED
          SIZE      CDATA       #IMPLIED
          SRC       %URL;       #IMPLIED
          TYPE      %InputType;     text
          VALUE     CDATA       #IMPLIED
          %input.addon; >
```

51.2 Description

The `<INPUT>` element is used in forms and represents a field for user input.

The start tag is required, but the end tag shall be omitted.

51.3 Attributes

The `TYPE` attribute of the `<INPUT>` element discriminates between several different types of input field. The set of applicable attributes depends on the value of the `TYPE` attribute as specified in the following subclauses. By default the value of the `TYPE` attribute is TEXT.

51.3.1 TYPE=CHECKBOX

An `<INPUT>` element with `TYPE=CHECKBOX` specifies a boolean choice. A set of `<INPUT>` elements in the same `<FORM>` element with the same `NAME` attribute value represents an n-of-many choice.

The other attribute values are as follows:

- CHECKED

  Used to mark the corresponding input item as initially selected.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  This attribute is required and specifies a name for this `<FORM>` field.

- VALUE

  This attribute is required and specifies the value to be returned if this input item is selected.

Example:

```
Which flavours do you like?
<INPUT TYPE=CHECKBOX NAME=flavour
       VALUE=van CHECKED>Vanilla <BR>
<INPUT TYPE=CHECKBOX NAME=flavour
       VALUE=str>Strawberry <BR>
<INPUT TYPE=CHECKBOX NAME=flavour
       VALUE=pec CHECKED>Pecan <BR>
```

Note that more than one flavour may be checked.

51.3.2 TYPE=FILE

An `<INPUT>` element with TYPE=FILE provides a means for users to attach a file to a form's contents. It is typically rendered by a text field and an associated button which when selected invokes a file browser to select a file name. The file name can also be entered directly in the text field. See RFC 1867 for further details [NM95].

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- MAXLENGTH

  The upper limit on the number of characters in the input field.

- NAME

  This attribute is required and specifies a name for this `<FORM>` field.

- SIZE

  Prefered size of the visible input line in average character widths. Users should be able to type more characters than this limit with the text scrolling through the field to keep the input cursor in view.

Example:

```
<FORM ACTION="http://server.rec/cgi/handle"
     ENCTYPE="multipart/form-data"
     METHOD=post>
What is your name?
<INPUT TYPE=TEXT NAME=submitter><BR>
Which file are you sending?
<INPUT TYPE=FILE NAME=pics>
</FORM>
```

The user types "Joe Sixpack" in the name field, and selects an image file "football.jpeg" for the answer to 'Which file are you sending?'.

51.3.3 TYPE=HIDDEN

An `<INPUT>` element with `TYPE=HIDDEN` declares that fields should not be rendered — they are hidden from the user. The user does not interact with the field; instead, the `VALUE` attribute specifies the value of the field. The `NAME` and `VALUE` attributes are required, and are returned to the server when the form is submitted.

Note 39 *This input element may be used to handle state information in a form.*

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  This attribute is required and specifies a name for this `<FORM>` field.

- VALUE

  This attribute is required and specifies a value for the field.

Example:

```
<INPUT TYPE=HIDDEN NAME=customer
                   VALUE="827364">
```

51.3.4 TYPE=IMAGE

An `<INPUT>` element with TYPE=IMAGE specifies an image resource to display. If a point in the image is selected, the x and y coordinates of that point are returned to the server. The names of the fields used for returning the x and y pixel coordinates are the specified name of the field with ".x" and ".y" appended.

TYPE=IMAGE implies TYPE=SUBMIT processing, that is when a pixel is chosen, the form as a whole is submitted.

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  This attribute is required and specifies a name for this <FORM> field.

- SRC

  This attribute is required and specifies the source of the image.

Example:

```
Choose a point on the map:
<INPUT TYPE=IMAGE SRC="world.jpeg"
       NAME=place>
```

When the user selects a point on the map, say with pixel coordinates (129,436), the input fields become place.x=129,place.y=436. See subclause 40.2.1 on page 31.

51.3.5 TYPE=PASSWORD

An <INPUT> element with TYPE=PASSWORD specifies a single line text field into which users may type a password. As the user types, the characters are usually echoed as '*' to hide the password from prying eyes.

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- MAXLENGTH

  The upper limit on the number of characters in the input field.

- NAME

  This attribute is required and specifies a name for this <FORM> field.

- SIZE

  Users should be able to type more characters than this limit with the password scrolling through the field to keep the input cursor in view.

Example:

```
Name:
<INPUT NAME=login>
Password:
<INPUT TYPE=PASSWORD SIZE=10
       MAXLENGTH=8 NAME=passwd>
```

51.3.6 TYPE=RADIO

An <INPUT> element with TYPE=RADIO specifies a boolean choice. A set of <INPUT> elements in a <FORM> element with the same NAME attribute value represents a 1-of-many choice.

The other attribute values are as follows:

- CHECKED

  Used to mark the corresponding input item as initially selected.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  This attribute is required and specifies a name for this <FORM> field.

- VALUE

  This attribute is required and specifies the value to be returned if this input item is selected.

At all times one and only one of the radio buttons in a set is checked. Initially, if none of the <INPUT> elements in a set of radio buttons specifies CHECKED, then the user agent shall mark the first radio button of the set as checked.

Example:

```
Which is your favourite flavour?
<INPUT TYPE=RADIO NAME=flavour
       VALUE=van>Vanilla <BR>
<INPUT TYPE=RADIO NAME=flavour
       VALUE=str>Strawberry <BR>
<INPUT TYPE=RADIO NAME=flavour
       VALUE=pec CHECKED>Pecan <BR>
```

51.3.7 TYPE=RESET

An <INPUT> element with TYPE=RESET specifies an input option, usually represented by a button, that instructs the user agent to reset the form's fields to their initial states.

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- VALUE

  Indicates a label for the input field. This value is usually displayed on a button.

Example:

```
When you are finished,
you may submit this request:
<INPUT TYPE=SUBMIT><BR>
You may clear the form
and start over at any time:
<INPUT TYPE=RESET VALUE="Reset">
```

51.3.8 TYPE=SUBMIT

An <INPUT> element with TYPE=SUBMIT represents an input option, typically a button, that instructs the user agent to submit the form. The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  Indicates that this element contributes a form field whose value is given by the VALUE attribute. If the NAME attribute is not present, this element does not contribute a form field.

- VALUE

  Indicates a label for the form field. This value is usually displayed on a button.

Example:

```
You may submit this request internally:
<INPUT TYPE=SUBMIT NAME=request
                   VALUE=internal>
or externally:
<INPUT TYPE=SUBMIT NAME=request
                   VALUE=external>
```

51.3.9 TYPE=TEXT

An <INPUT> element with TYPE=TEXT specifies a single line text field into which users may type a string.

The other attribute values are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- MAXLENGTH

  The upper limit on the number of characters in the input field.

- NAME

  This attribute is required and specifies a name for this <FORM> field.

- SIZE

  Prefered size of the visible input line in average character widths. Users should be able to type more than this limit with the text scrolling through the field to keep the input cursor in view.

- VALUE

  This attribute is required and specifies an initial value for the text string initially shown in the field.

Example:

```
<INPUT TYPE=TEXT SIZE=40 NAME=user
VALUE="Intentionally left blank">
```

51.4 Example

For examples of the use of the <INPUT> element, please see subclauses 51.3.1 on page 38, 51.3.2 on page 39, 51.3.3 on page 39, 51.3.4 on page 39, 51.3.5 on page 40, 51.3.6 on page 40, 51.3.7 on page 41, 51.3.8 on page 41 and 51.3.9 on page 41.

52 The ISINDEX element — Keyword choice

52.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT ISINDEX  - O  EMPTY>
<!ATTLIST ISINDEX
          CLASS NAMES   #IMPLIED
          DIR   %dirn;  #IMPLIED
          ID    ID      #IMPLIED
          LANG  NAME    #IMPLIED
          PROMPT CDATA  #IMPLIED
          %isindex.addon; >
```

52.2 Description

The <ISINDEX> element implies a set of hyperlinks. A user agent enables users to choose from this set by providing keywords.

Note 40 *User agents compute target URLs by appending the character '?' and the selected keywords to the base URL. Keywords are escaped according to RFC 1738 [BLMM94] and joined together by the character '+'.*

<FORM> elements with METHOD=get also represent sets of hyperlinks. See clause 40 on page 31 for details.

The start tag is required, but the end tag shall be omitted.

52.3 Attributes

The <ISINDEX> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- PROMPT

  A text which encourages a user to make a choice.

52.4 Example

If a document <HEAD> element contains:

```
<BASE HREF="http://host/index">
<ISINDEX>
```

and a user selects the keywords apple and berry, then the user agent will access the resource

```
http://host/index?apple+berry
```

53 The ISO-HTML element — Document instance

53.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT ISO-HTML  - -  (HEAD, BODY) >
<!ATTLIST ISO-HTML
          CLASS NAMES    #IMPLIED
          DIR   %dirn;   #IMPLIED
          ID    ID       #IMPLIED
          LANG  NAME     #IMPLIED
          VERSION CDATA    #FIXED "CD"
          %iso-html.addon; >
```

53.2 Description

The <ISO-HTML> element contains a document instance marked up in the language specified by this Committee Draft.

Both start and end tags are required.

53.3 Attributes

The <ISO-HTML> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- VERSION

  The VERSION attribute of the <ISO-HTML> document element may be omitted, but if specified shall have the value CD.

53.4 Example

```
<ISO-HTML>
<HEAD>
<TITLE>Scientific Units</TITLE>
```

... *other head elements*

```
</HEAD>
<BODY>
<H1>Scientific units and their conversions<H1>
```

... *document body*

```
</BODY>
</ISO-HTML>
```

54 The KBD element — Keyboard input

54.1 Formal definition

```
<!ENTITY % logical.styles
           "... | KBD | ..." >
<!ENTITY % dirn  "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

54.2 Description

The <KBD> element contains text to be typed by a user.

Both start and end tags are required.

54.3 Attributes

The <KBD> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

54.4 Example

```
At the prompt, you might try typing:
<KBD>rm -rf *</KBD>
```

55 The LI element — List item

55.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT LI  - O  %section.content; >
<!ATTLIST LI
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          VALUE NUMBER    #IMPLIED
          %li.addon; >
```

55.2 Description

The <LI> element contains a list item in an ordered list [clause 61 on page 48] or an unordered list [clause 83 on page 61].

The start tag is required, but the end tag may be omitted.

55.3 Attributes

The <LI> element has the following attributes:

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- VALUE

  This attribute is optional and shall only be used in ordered lists. It provides an integer number for the ordered list item and re-sequences the numbering of the items in the ordered list.

55.4 Example

see subclause 61 on page 48 and 83 on page 61 for examples of the use of the <LI> element.

56 The LINK element — Interdocument relations

56.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT LINK  - O  EMPTY >
<!ATTLIST LINK
          CHARSET NAME    #IMPLIED
          CLASS   NAMES   #IMPLIED
          DIR     %dirn;  #IMPLIED
          HREF    %URL;   #IMPLIED
          ID      ID      #IMPLIED
          LANG    NAME    #IMPLIED
          REL     CDATA   #IMPLIED
          REV     CDATA   #IMPLIED
          TITLE   CDATA   #IMPLIED
          %link.addon; >
```

56.2 Description

The <LINK> element provides a means for specifying relationships between this hypertext document and other documents and resources.

Any number of <LINK> elements may appear in the head of a document and are typically used to indicate authorship, related indexes and glossaries, older and more recent versions, document hierarchy and associated resources such as style sheets.

The start tag is required, but the end tag shall be omitted.

56.3 Attributes

The <LINK> element has the following attributes:

- CHARSET

  Please see the CHARSET attribute of the <A> element [clause 15 on page 19].

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- HREF

  Please see the HREF attribute of the <A> element.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- REL

  Please see the REL attribute of the <A> element.

- REV

  Please see the REV attribute of the <A> element.

- TITLE

  Please see the TITLE attribute of the <A> element.

56.4 Example

See clause 53 on page 42 for an example of the use of the <LINK> element.

57 The MAP element — Client-side image map

57.1 Formal definition

```
<!ELEMENT MAP  - -  (AREA)+ >
<!ATTLIST MAP
          NAME CDATA #REQUIRED
          %map.addon; >
```

57.2 Description

The <MAP> element provides a mechanism for constructing client-side image maps. It acts as a container for a set of <AREA> elements which define individual regions in an image [clause 17 on page 21].

Within a <MAP> careful attention should be given to the ordering of the <AREA> elements: If two or more regions overlap, then the first <AREA> element listed will take precedence over subsequent <AREA> elements, etc. Thus <AREA> elements with no associated action (which have the NOHREF attribute), should be specified in the <MAP> before <AREA>s defining hotzones (which have the HREF attribute).

Both start and end tags are required.

57.3 Attributes

The <MAP> element has the following attribute:

- NAME

  This required attribute specifies a name for the map which may be referenced by the USEMAP attribute of an <IMG> element using a URL fragment identifier 50 on page 37.

  Note 41 *The value of the NAME attribute is case sensitive.*

57.4 Example

```
<IMG SRC="USA.jpeg" USEMAP="#States">
...
<MAP NAME="States">
   ...
   <AREA HREF="Florida.html" ALT="Florida"
         SHAPE=poly COORDS="...">
   ...
   <AREA HREF="Texas.html" ALT="Texas"
         SHAPE=poly COORDS="...">
   ...
</MAP>
```

58 The META element — Meta-information

58.1 Formal definition

```
<!ELEMENT META  - O  EMPTY >
<!ATTLIST META
          CONTENT    CDATA #REQUIRED
          HTTP-EQUIV NAME   #IMPLIED
          NAME       NAME   #IMPLIED
          %meta.addon; >
```

58.2 Description

The <META> element is an extensible container for use in identifying specialized document meta-information. Meta-information has two main functions:

- To provide a means to verify that the data set exists, and determine how it might be obtained or accessed.
- To document the content, quality and features of a data set, indicating its fitness for use.

Each <META> element specifies a name/value pair. If multiple <META> elements are provided with the same name, their combined contents — concatenated as a comma-separated list — is the value that will be associated with that name.

Note 42 *The <META> element should not be used where a specific element, such as <TITLE>, would be more appropriate. Rather than specifying a <META> element with a URL as the value of the CONTENT attribute, use a <LINK> element instead.*

Note 43 *HTTP servers may read the content of a document's <HEAD> element to generate header fields corresponding to those elements defining a value for the attribute HTTP-EQUIV*

Note 44 *The method by which the server extracts document meta-information is unspecified and not mandatory. The <META> element only provides an extensible mechanism for identifying and embedding such document meta-information — how it may be used is up to the individual server implementation and ISO-HTML user agent.*

An HTTP server shall not use the <META> element to form an HTTP response header unless the HTTP-EQUIV attribute is present.

An HTTP server may disregard any <META> elements that specify information controlled by the HTTP server, for example 'Server', 'Date', and 'Last-modified'.

The start tag is required, but the end tag shall be omitted.

58.3 Attributes

The attributes of the `<META>` element are:

- CONTENT

  This attribute is required and specifies the value part of a name/value pair.

- HTTP-EQUIV

  Binds the element to an HTTP header field. An HTTP server may use this information to process the document. In particular, it may include a header field in the responses to requests for this document: the header name is taken from the `HTTP-EQUIV` attribute value, and the header value is taken from the value of the `CONTENT` attribute. HTTP header names are not case sensitive.

- NAME

  Specifies the name part of the name/value pair. If not present, then the `HTTP-EQUIV` attribute is used to specify the name.

Either the `HTTP-EQUIV` or the `NAME` attribute shall be present.

58.4 Example

If a document contains:

```
<META HTTP-EQUIV="Expires"
      CONTENT="Sat, 1 Jan 2000 00:00:01 UTC">
<META HTTP-EQUIV="Keywords" CONTENT="Fred">
<META HTTP-EQUIV="Reply-to"
      CONTENT="Flinstones@bedrock.old">
<META HTTP-EQUIV="Keywords" CONTENT="Barney">
```

then the server may include the following header fields:

```
Expires: Sat, 1 Jan 2000 00:00:01 UTC
Keywords: Fred, Barney
Reply-to: Flintstones@bedrock.old
```

as part of an HTTP response to a '`get`' or '`post`' request for that document.

59 The NOP element — Interface default value

59.1 Formal definition

```
<!ELEMENT NOP  - O  EMPTY >
```

59.2 Description

The `<NOP>` element has no content and no meaning. It shall be ignored by user agents, and shall not be used in conforming documents.

Note 45 *The `<NOP>` element is used in the DTD to ensure syntactically correct content models for unused interfaces.*

59.3 Attributes

The `<NOP>` element has no attributes.

60 The OBJECT element — Simple agent

60.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT OBJECT  - -
          (PARAM | %section.content;)* >
<!ATTLIST OBJECT
          ALIGN
   (texttop,middle,textmiddle,baseline,
    textbottom,left,center,centre,right)
                              #IMPLIED
          CLASS     NAMES     #IMPLIED
          CLASSID   %URL;     #IMPLIED
          CODEBASE  %URL;     #IMPLIED
          CODETYPE  CDATA     #IMPLIED
          DATA      %URL;     #IMPLIED
          DECLARE   (declare) #IMPLIED
          DIR       %dirn;    #IMPLIED
          ID        ID        #IMPLIED
          LANG      NAME      #IMPLIED
          NAME      %URL;     #IMPLIED
          STANDBY   CDATA     #IMPLIED
          TYPE      CDATA     #IMPLIED
          USEMAP    %URL;     #IMPLIED
          %object.addon; >
```

Note 46 Editors to NB experts: *The `NAME` attribute is declared as having type `%URL;`. Is this correct? [Dix points].*

60.2 Description

The `<OBJECT>` element provides a technique for specifying richer behaviour for the document in the form of a simple agent. The scheme uses only elements and attributes and is simple to implement.

Note 47 *This technique deliberately avoids using sophisticated SGML techniques in order to facilitate its use by current user agents. Nothing in this Committee Draft prevents* *the use of those SGML techniques by more sophisticated user agents.*

The specification of the `<OBJECT>` element supports the general insertion of media other than text into documents, but excludes the agent architecture, application programming interface (API) and inter-agent communication.

The agent may access data specified:

- In the document text flow,
- As a set of named properties,
- In a separate file which is the target of a link specified within the document,
- Using techniques outside the scope of this Committee Draft.

A conforming ISO-HTML browser is not required to support the agent behaviour, and if it does not it shall render the `%section.content;` part of the content. Browsers which render the agent behaviour shall not render the `%section.content;`. If a browser chooses to render the `%section.content;`, then it may also display the content of the `ALT` attribute to indicate that this choice has been made.

It is recommended that authors provide a suitable substitute for the agent behaviour in the `%section.content;` part of the content of the element.

The agents are normally loaded relative to the document URL (or `<BASE>` element if it is defined), and the `CODEBASE` attribute may be used to change this default behaviour. If the `CODEBASE` attribute is defined, then it specifies a different location for the agent resources. The value can be an absolute or relative URL. An absolute URL is used as is without modification and is not effected by a document's `<BASE>` element. When the `CODEBASE` attribute is relative, then it is relative to the document URL (or `<BASE>` tag if defined).

Both start and end tags are required.

60.3 Attributes

The `<OBJECT>` element has the following attributes:

- ALIGN

  Please see subclause 14 on page 19.

- CLASS

  Please see subclause 9.2 on page 14.

- CLASSID

  Provides a URL which identifies an implementation for the agent. In some systems the value is a class identifier. The values are not defined by this Committee Draft. If this attribute is omitted, a default value may be deduced from the `DATA` attribute value.

Note 48 *Since the class identifiers in some object systems may be cumbersome, the `CLASSID` attribute may use a short URL to specify a class identifier indirectly.*

- CODEBASE

  Some URL schemes used to identify implementations require an additional URL to locate the implementation. This attribute provides the additional URL. The values are not defined by this Committee Draft.

- CODETYPE

  The Internet media type of the code referenced by the `CLASSID` attribute.

- DATA

  Provides a URL for the agent's data, eg. a JPEG file. The `TYPE` attribute may provide the Internet Media Type for the data. In the absence of the `CLASSID` attribute, the Internet Media Type of the data gives the default value to the `CLASSID` attribute.

- DECLARE

  The `DECLARE` attribute calls for late binding of the agent. The attribute is usually used to indicate that

  - `<PARAM>` elements are to be "object valued".
  - Hyperlinks point to objects which cannot otherwise be addressed with a single URL.

  Each such binding typically results in a separate instantiation of the agent. `<OBJECT DECLARE>` is treated as a declaration for making an instance of the agent.

  If the declared agent is not supported, the user agent should use the content of the `<OBJECT DECLARE>` element which may contain another `<OBJECT>` element. The `TYPE` attribute may be used to specify the Internet Media Type for the outer agent as a hint for this situation.

  Anchors may exploit nested declared agents to provide alternative media for a given resource.

Note 49 Editors to NB experts: *This explanation of the `DECLARE` attribute is based on text that came from the W3C. It needs further work and some good examples.*

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14. The ID attribute is also available to assist inter agent communication.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  If the NAME attribute is present and if the <OBJECT> element is contained in a <FORM> element, then this agent participates in the form submission process. If NAME is present and the DECLARE attribute is absent, the user agent should include the value of the NAME attribute and the data obtained from the agent along with information derived from other form fields, see subclause 40.2.1 on page 31.

  Note 50 *The mechanism used to obtain an agent's data is specific to each agent.*

- STANDBY

  The attribute value provides a message to be displayed while the agent is being prepared.

- TYPE

  The Internet Media Type of the data referenced through use of the DATA attribute.

- USEMAP

  Specifies a URL fragment identifier for a client-side image map defined with the <MAP> element, see subclause 57 on page 45. In general, the use of this attribute is only appropriate for static images.

60.4 Example

Note 51 Editors to NB experts: *Good examples are urgently needed.*

61 The OL element — Ordered list

61.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT (OL|UL)  - -  (LI)+ >
<!ATTLIST OL
        ALIGN   %justify; #IMPLIED
        CLASS   NAMES     #IMPLIED
        COMPACT (compact) #IMPLIED
        DIR     %dirn;    #IMPLIED
        ID      ID        #IMPLIED
        LANG    NAME      #IMPLIED
        START   NUMBER    #IMPLIED
        %ol.addon; >
```

61.2 Description

The <OL> element represents a non empty ordered list of items, presented in sequence or order of importance. It is usually rendered as a numbered list.

The content is a sequence of <LI> elements [clause 55 on page 43].

Both start and end tags are required.

61.3 Attributes

The attributes of the <OL> element are as follows:

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- COMPACT

  The optional COMPACT attribute suggests that a compact rendering be used, perhaps because the list items are small, or the entire list is large. Unless the COMPACT attribute is present, a user agent may leave white space between successive <LI> elements.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- START

  Specifies an initial value for the item sequence number. If the START attribute is omitted, the default start value is 1. If specified, the value shall be an integer.

61.4 Example

```
Easter Calendar
<OL START=1997>
<LI>30th March
<LI>12th April
<LI>4th April
<LI>23rd April
</OL>
```

62 The OPTION element — User choice

62.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT OPTION  - O  (#PCDATA) >
<!ATTLIST OPTION
          CLASS NAMES         #IMPLIED
          DIR   %dirn;        #IMPLIED
          ID    ID            #IMPLIED
          LANG  NAME          #IMPLIED
          SELECTED (selected) #IMPLIED
          VALUE    CDATA      #IMPLIED
          %option.addon; >
```

62.2 Description

The `<OPTION>` element can only occur within a `<SELECT>` element and represents one of the possible choices that is offered. The content of the `<OPTION>` element is presented to users as a representation of the option. Its content is used as the returned value if the `VALUE` attribute is not present.

The start tag is required, but the end tag may be omitted.

62.3 Attributes

The attributes of the `<OPTION>` element are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- SELECTED

  Indicates that the option is initially selected.

- VALUE

  Indicates the value to be returned if this option is chosen. If omitted, the returned value defaults to the content of the `<OPTION>` element.

  Note 52 *The name associated with this value is given by the `NAME` attribute of the parent `<SELECT>` element.*

62.4 Example

See clause 68 on page 52 for an example of the use of the `<OPTION>` element.

63 The P element — Paragraph

63.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT P  - O  (%text;)+ >
<!ATTLIST P
          ALIGN %justify; #IMPLIED
          CLASS NAMES     #IMPLIED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          %p.addon; >
```

63.2 Description

The contents of the `<P>` element form a paragraph.

Note 53 *Authors often start paragraphs on new lines to facilitate document maintenance.*

The start tag is required, but the end tag may be omitted.

63.3 Attributes

The `<P>` element has the following attributes:

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

63.4 Example

```
<P>
It was a dark and stormy night ...
```

64 The PARAM element — Agent interface parameter

64.1 Formal definition

```
<!ELEMENT PARAM - O EMPTY>
<!ATTLIST PARAM
          NAME  NAME  #REQUIRED
          TYPE  CDATA  #IMPLIED
          VALUE CDATA  #IMPLIED
          VALUETYPE
            (data|ref|object) data
          %param.addon; >
```

64.2 Description

The <PARAM> element appears only in <OBJECT> element content and specifies the name and value of a parameter to be passed to the agent [clause 60 on page 46]. The <PARAM> element has no content.

The start tag is required, but the end tag shall be omitted.

64.3 Attributes

The <PARAM> element has the following attributes:

- NAME

  This attribute is required and specifies the name of the parameter.

  Note 54 *Case sensitivity is implementation dependent.*

- TYPE

  This attribute is used only when the `VALUETYPE` attribute has the value "`ref`" and specifies the Internet Media Type of the referenced value.

- VALUE

  Specifies the value of the parameter. For example: `VALUE="expensive"`.

- VALUETYPE

  Specifies how the `VALUE` attribute is to be interpreted. Possible values are as follows:

  – `VALUETYPE=data`

    The value "expensive" is to be passed directly to the agent as a string. In the absence of a `VALUETYPE` attribute, "data" is the default behaviour for the `VALUETYPE` attribute.

  – `VALUETYPE=object`

    The value "expensive" is treated as the URL of an object or agent in the same document. This is used primarily for object or agent valued properties where the value of a property is a pointer (or reference) to a running object or to an agent.

  – `VALUETYPE=ref`

    The value "expensive" is a URL.

  Note 55 *The attribute value is obtained from the attribute value literal after replacing any entity references or character references within the literal and then normalizing the result by throwing out any entity ends and record starts and replacing any record end or separator characters with a space [see Goldfarb p.331].*

64.4 Example

See clause 60 on page 46 for an example of the use of the <PARAM> element.

65 The PRE element — Pre-formatted text

65.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT PRE  - -  (%text;)+
   -(BIG|IMG|MAP|SMALL|SUB|SUP|%body.extend;) >
<!ATTLIST PRE
          CLASS NAMES   #IMPLIED
          DIR   %dirn;  #IMPLIED
          ID    ID      #IMPLIED
          LANG  NAME    #IMPLIED
          WIDTH NUMBER  #IMPLIED
          %pre.addon; >
```

65.2 Description

The <PRE> element contains a non-empty block of text that is to be formatted as is, with its contents rendered in a mono-spaced font.

Within pre-formatted text, line breaks are rendered faithfully as line breaks. A user agent may use a constant indent when rendering pre-formatted text.

Anchor elements and phrase markup may be embedded in pre-formatted text.

Note 56 *Constraints on the processing of* `<PRE>` *element content may limit the ability of a user agent to faithfully render phrase markup.*

Both start and end tags are required.

65.3 Attributes

The `<PRE>` element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.
- WIDTH

  Specifies the maximum number of characters in a line and allows a user agent to select a suitable font and indentation. The value of this attribute shall be a strictly positive integer.

65.4 Example

```
<PRE WIDTH=16>
  Tic Tac Toe
   X |   | O
  -----------
   O | X |
  -----------
     |   | X
</PRE>
```

66 The Q element — Quote

66.1 Formal definition

```
<!ENTITY % logical.styles "... | Q | ..." >
<!ENTITY % dirn              "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

66.2 Description

The `<Q>` element contains a quotation. Rendering is usually language and platform dependent.

Both start and end tags are required.

66.3 Attributes

The `<Q>` element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.

66.4 Example

```
A <Q LANG=de>quotation in German</Q>
and a <Q LANG=fr>quotation in French</Q>.
```

might be rendered as:

A „quotation in German" and a ≪ quotation in French ≫.

67 The SAMP element — Sample output

67.1 Formal definition

```
<!ENTITY % logical.styles
           "... | SAMP | ..." >
<!ENTITY % dirn   "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

67.2 Description

The <SAMP> element contains sample output from a program or a script.

Both start and end tags are required.

67.3 Attributes

The <SAMP> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

67.4 Example

```
The installation begins:
<samp>Insert diskette 1</samp>
```

68 The SELECT element — Selection in forms

68.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT SELECT  - -  (OPTION)+ >
<!ATTLIST SELECT
          CLASS    NAMES       #IMPLIED
          DIR      %dirn;      #IMPLIED
          ID       ID          #IMPLIED
          LANG     NAME        #IMPLIED
          MULTIPLE (multiple)  #IMPLIED
          NAME     CDATA      #REQUIRED
          SIZE     NUMBER      #IMPLIED
          %select.addon; >
```

68.2 Description

The <SELECT> element defines one-from-many and many-from-many menus. The items from which the choice is made are defined by a sequence of <OPTION> elements [clause 62 on page 49].

At least one <OPTION> shall be specified in a <SELECT> element.

One-from-many menus are usually rendered as drop-down menus while many-from-many menus are usually rendered as list menus.

The initial state has the first option selected, unless a SELECTED attribute is present on at least one of the <OPTION> elements [clause 62 on page 49].

Both start and end tags are required.

68.3 Attributes

The <SELECT> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- MULTIPLE

  The presence of this attribute signifies that the users can make multiple selections. If this attribute is omitted, at most one selection is allowed.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- NAME

  Specifies the name used to identify the menu choice when the form is submitted to a server. Each selected option results in a name/pair value being included in the form's contents. This attribute is required.

- SIZE

  Suggests the number of visible choices.

68.4 Example

```
<SELECT NAME="flavour">
   <OPTION>Vanilla
   <OPTION SELECTED>Chocolate
   <OPTION VALUE="LemonLime">Lemon and Lime
   <OPTION>Strawberry
</SELECT>
```

69 The SMALL element — Small character style

69.1 Formal definition

```
<!ENTITY % physical.styles
           "... | SMALL | ..." >
<!ENTITY % dirn    "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

69.2 Description

The contents of a <SMALL> element should be rendered in a smaller font than would otherwise be used.

Both start and end tags are required.

69.3 Attributes

The <SMALL> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

69.4 Example

```
<BIG>What the large print giveth,</BIG>
<SMALL>the small print taketh away.</SMALL>
```

70 The SPAN element — Generic container

70.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT SPAN  - -  (%text;)+ >
<!ATTLIST SPAN
          CLASS NAMES   #IMPLIED
          DIR   %dirn;  #IMPLIED
          ID    ID      #IMPLIED
          LANG  NAME    #IMPLIED
          %span.addon >
```

70.2 Description

The <SPAN> element provides a generic container to carry language and bidirectional attributes in cases where no other element is appropriate.

Both start and end tags are required.

70.3 Attributes

The optional attributes of the <SPAN> element are as follows:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

70.4 Example

```
<P>
The licence plate
<SPAN DIR=ltr>CIA 1</SPAN>
will appear correctly even if cut
and pasted into the source of a report
normally rendered in a right-to-left
fashion.
```

71 The STRIKE element — Strike-out character style

71.1 Formal definition

```
<!ENTITY % physical.styles
           "... | STRIKE | ..." >
<!ENTITY % dirn    "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
```

```
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

71.2 Description

The contents of a <STRIKE> element should be rendered in a "strike-through" text style if available, otherwise an alternative representation may be used.

Both start and end tags are required.

71.3 Attributes

The <STRIKE> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.

71.4 Example

```
The original description should
<STRIKE>NOT</STRIKE> have been
omitted.
```

72 The STRONG element — Strong emphasis

72.1 Formal definition

```
<!ENTITY % logical.styles
           "... | STRONG | ..." >
<!ENTITY % dirn     "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

72.2 Description

The contents of a <STRONG> element are strongly emphasised and are usually rendered in a bold font.

Both start and end tags are required.

72.3 Attributes

The <STRONG> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.

72.4 Example

```
<EM>High Voltage <STRONG>Danger</STRONG></EM>
```

73 The SUB element — Subscript character style

73.1 Formal definition

```
<!ENTITY % physical.styles
           "... | SUB | ..." >
<!ENTITY % dirn  "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

73.2 Description

A <SUB> element contains subscripted text. Its contents should be positioned below the base line and rendered in a smaller size than the preceding text.

Both start and end tags are required.

73.3 Attributes

The <SUB> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

73.4 Example

```
H<SUB>2</SUB>O
```

74 The SUP element — Superscript character style

74.1 Formal definition

```
<!ENTITY % physical.styles
           "... | SUP | ..." >
<!ENTITY % dirn  "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

74.2 Description

A <SUP> element contains superscripted text. Its contents should be positioned above the base line and rendered in a smaller size than the preceding text.

Both start and end tags are required.

74.3 Attributes

The <SUP> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

74.4 Example

```
E = mc<SUP>2</SUP>
```

75 The TABLE element — Tables

75.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT TABLE  - -  (CAPTION?, TR+) >
<!ATTLIST TABLE
          ALIGN
        (left,center,centre,right) left
          BORDER      %pixels; #IMPLIED
          CELLPADDING %pixels; #IMPLIED
          CELLSPACING %pixels; #IMPLIED
          CLASS       NAMES    #IMPLIED
          DIR         %dirn;   #IMPLIED
          DUMMY       (border) #IMPLIED
          ID          ID       #IMPLIED
          LANG        NAME     #IMPLIED
          WIDTH       %length; #IMPLIED
          %table.addon; >
```

75.2 Description

The <TABLE> element provides a widely deployed subset of the specification given in RFC 1942 [Rag96]. It is used to

- Markup tabular material.
- Describe tabular layout.

> **Note 57** *Authors should be aware that this latter use causes problems when rendering to speech or text-only agents.*

A table contains an optional caption [clause 31 on page 27] followed by a non-empty sequence of table rows [clause 80 on page 59].

Both start and end tags are required.

75.3 Attributes

The attributes of the <TABLE> element are as follows:

Note 58 *The pixel values refer to screen pixels, and should be multiplied by an appropriate factor when rendering to high resolution output devices such as laser printers. For instance if a user agent has a display with 75 pixels per inch and is rendering to a laser printer with 600 dots per inch, then the pixel values given in the attributes should be multiplied by a factor of 8 prior to output.*

- ALIGN

  Please see subclause 14 on page 19.

- BORDER

  Specifies the width of the outer border around a table in screen pixels, eg. `BORDER=4`. The border may be suppressed by setting `BORDER=0`. If both this attribute and the attribute `DUMMY` are omitted, the border should be suppressed.

- CELLPADDING

  Specifies the padding in screen pixels between a cell's contents and the border around the cell.

- CELLSPACING

  Each cell in a table has its own border which is separated from the borders around neighbouring cells. This separation can be set using the `CELLSPACING` attribute, eg. `CELLSPACING=10` for a 10 pixel separation. This attribute also determines the separation between the table border and the borders of the outermost cells.

  Note 59 *In traditional desktop publishing software, adjacent table cells share a common border.*

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- DUMMY

  Allows an author to specify the attribute value BORDER as if it were an attribute name with no associated value. Such a specification requests that the border be drawn using the user agent's default width, usually one pixel.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- WIDTH

  Sets the table width. If an integer is specified, the unit is the screen pixel, eg. `WIDTH=212`, but if a percentage is specified by concatenating a "`%`" character as a suffix to the value, the table width is defined as a percentage of the space between the current left and right margins, eg. `WIDTH="80%"`.

  If this attribute is omitted, the table width is automatically determined from the table contents.

75.4 Example

Tables take the general form:

```
<TABLE BORDER=3 WIDTH="64%"
       CELLSPACING=2 CELLPADDING=2>
<CAPTION>Complexity of
         sorting algorithms</CAPTION>
<TR><TH> Algorithm      <TH> Complexity
<TR><TD> Random shuffle <TD> O(exp(N))
<TR><TD> Bubble sort    <TD> O(NxN)
<TR><TD> Quicksort      <TD> O(NlogN)
<TR><TD> Heap sort      <TD> O(NlogN)
</TABLE>
```

76 The TD element — Table data cell

76.1 Formal definition

```
<!ENTITY % dirn              "(ltr|rtl)" >
<!ENTITY % valign
          "(top|middle|bottom|baseline)" >
<!ELEMENT (TH|TD)  - O  %table.content; >
<!ATTLIST ( ... | TD )
          ALIGN   %justify; #IMPLIED
          CLASS   NAMES     #IMPLIED
          COLSPAN NUMBER           1
          DIR     %dirn;    #IMPLIED
          ID      ID        #IMPLIED
          LANG    NAME      #IMPLIED
          NOWRAP  (nowrap)  #IMPLIED
          ROWSPAN NUMBER           1
          VALIGN  %valign;  #IMPLIED
          %td.addon; >
```

76.2 Description

The `<TD>` element is found only in `<TR>` table rows [clause 80 on page 59] and contains a data cell in a table.

The start tag is required, but the end tag may be omitted.

76.3 Attributes

The attributes of the `<TD>` element are as follows:

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- COLSPAN

  Specifies the number of columns spanned by this cell. The value is a strictly positive integer, and if not specified, defaults to 1.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NOWRAP

  The presence of this attribute disables automatic word wrap within the contents of the cell.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

  Note 60 *This is equivalent to using the   entity for non-breaking spaces within the content of the cell.*

- ROWSPAN

  Specifies the number of rows spanned by this cell. The value is a strictly positive integer, and if not specified, defaults to 1.

- VALIGN

  Specifies a default vertical alignment for cell contents which overrides any value specified by the containing `<TR>` element [clause 80 on page 59]. The permitted values are "`top`", "`middle`", "`bottom`" and "`baseline`". If not specified, the default vertical alignment is "`middle`", unless overridden by the containing `<TR>` element. The meanings of the permitted values are the same as those described for the `ALIGN` attribute in subclause 14 on page 19.

Note 61 *A table cell may contain a nested table.*

76.4 Example

See clause 75 on page 55 for an example of the use of the `<TD>` element.

77 The TEXTAREA element — Multi-line text field

77.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT TEXTAREA  - -  (#PCDATA) >
<!ATTLIST TEXTAREA
          CLASS NAMES     #IMPLIED
          COLS  NUMBER   #REQUIRED
          DIR   %dirn;    #IMPLIED
          ID    ID        #IMPLIED
          LANG  NAME      #IMPLIED
          NAME  CDATA    #REQUIRED
          ROWS  NUMBER   #REQUIRED
          %textarea.addon; >
```

77.2 Description

The `<TEXTAREA>` element is used in forms to represent the initial value of a multi-line text field which may be modified by a user. Its contents are restricted to text and character entities.

Note 62 *It is recommended that user agents canonicalize line endings to CRLF when submitting a field's contents to a server.*

Note 63 *This Committee Draft does not specify the coded character set to be used for submitting such text to a server.*

The field is usually rendered in a mono-spaced font. Both start and end tags are required.

77.3 Attributes

The attributes of the `<TEXTAREA>` element are:

- CLASS

  Please see subclause 9.2 on page 14.

- COLS

  This required attribute specifies the prefered width of the field in visible mono-spaced characters. Users should be able to enter longer lines than this, so user agents should provide a means for scrolling through the text field when its contents extend beyond the visible area. User agents may wrap visible text lines to keep long lines visible without the need for scrolling.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NAME

  This required attribute specifies the name that is used to identify the field when the form is submitted to a server.

- ROWS

  This required attribute specifies the number of visible text lines. Users should be able to enter more lines that this, so user agents should provide some means for scrolling through the text field when its contents extend beyond the visible area.

77.4 Example

```
<TEXTAREA NAME=comments ROWS=3 COLS=32>
I found that your product was excellent!
</TEXTAREA>
```

78 The TH element — Table header

78.1 Formal definition

```
<!ENTITY % dirn             "(ltr|rtl)" >
<!ENTITY % valign
          "(top|middle|bottom|baseline)" >
<!ELEMENT (TH|TD)  - O  %table.content; >
<!ATTLIST ( TH | ... )
          ALIGN   %justify;#IMPLIED
          CLASS   NAMES    #IMPLIED
          COLSPAN NUMBER          1
          DIR     %dirn;   #IMPLIED
          ID      ID       #IMPLIED
          LANG    NAME     #IMPLIED
          NOWRAP  (nowrap) #IMPLIED
          ROWSPAN NUMBER          1
          VALIGN  %valign; #IMPLIED
          %th.addon; >
```

78.2 Description

The <TH> element is found only in <TR> table rows [clause 80 on page 59] and contains a header cell in a table. <TH> elements are distinguished from <TD> elements so that a user agent may provide different rendering styles for cell header and cell data.

<TH> is otherwise identical to <TD> except that if the ALIGN attribute is not specified, the default alignment is "centre" unless overridden by the enclosing <TR> element.

The start tag is required, but the end tag may be omitted.

78.3 Attributes

The <TH> element has the following attributes:

- ALIGN

  Please see subclause 9.1 on page 13 for a general description. This use of the ALIGN attribute differs only in that the default value "centre" is to be used if none is supplied.

- CLASS

  Please see subclause 9.2 on page 14.

- COLSPAN

  Specifies the number of columns spanned by this cell. The value is a strictly positive integer, and if not specified, defaults to 1.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

- NOWRAP

  The presence of this attribute disables automatic word wrap within the contents of the cell.

  Note 64 *This is equivalent to using the   entity for non-breaking spaces within the content of the cell.*

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- ROWSPAN

  Specifies the number of rows spanned by this cell. The value is a strictly positive integer, and if not specified, defaults to 1.

- VALIGN

  Specifies a default vertical alignment for cell contents which overrides any value specified by the containing <TR> element [clause 80 on page 59]. The permitted values are "`top`", "`middle`", "`bottom`" and "`baseline`". If not specified, the default vertical alignment is "`middle`", unless overridden by the containing <TR> element. The meanings of the permitted values are the same as those described for the `ALIGN` attribute in subclause 14 on page 19.

78.4 Example

See clause 75 on page 55 for an example of the use of the <TH> element.

79 The TITLE element — Document title

79.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT TITLE  - -  (#PCDATA)
  -(LINK | META | %head.extend;) >
<!ATTLIST TITLE
          CLASS NAMES   #IMPLIED
          DIR   %dirn;  #IMPLIED
          ID    ID      #IMPLIED
          LANG  NAME    #IMPLIED
          %title.addon; >
```

Note 65 Editors to NB experts: *Shouldn't the content model be replaced by RCDATA?*

79.2 Description

The <TITLE> element identifies the contents of a document in a global context. A short title such as "Introduction" may be meaningless out of context, and hence a title such as "Introduction to medieval bee-keeping" might be more appropriate.

Note 66 *The length of a title is not limited; however long titles may be truncated in some applications. To minimize this possibility titles should contain fewer than 64 characters.*

An ISO-HTML document shall contain one <TITLE> element.

A user agent may display the title of a document in a history list or as a label for the window displaying the document's content.

Both start and end tags are required.

79.3 Attributes

The <TITLE> element has the following attributes:

- CLASS

  Please see subclause 9.2 on page 14.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

79.4 Example

See clause 53 on page 42 for an example of the use of the <TITLE> element.

80 The TR element — Table row

80.1 Formal definition

```
<!ENTITY % dirn              "(ltr|rtl)" >
<!ENTITY % valign
          "(top|middle|bottom|baseline)" >
<!ELEMENT TR  - O  (TH|TD)+ >
<!ATTLIST TR
          ALIGN  %justify; #IMPLIED
          CLASS  NAMES     #IMPLIED
          DIR    %dirn;    #IMPLIED
          ID     ID        #IMPLIED
          LANG   NAME      #IMPLIED
          VALIGN %valign;  #IMPLIED
          %tr.addon; >
```

80.2 Description

The <TR> is found only in a <TABLE> element [clause 75 on page 55] and specifies a non-empty row in a table. It acts as a container for <TH> table header [clause 78 on page 58] and <TD> table data [clause 76 on page 56] elements.

The start tag is required, but the end tag may be omitted.

80.3 Attributes

The <TR> element has the following attributes:

- ALIGN

  Please see subclause 9.1 on page 13.
- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.
- VALIGN

  Specifies a default vertical alignment for cell contents. The permitted values are "top", "middle", "bottom" and "baseline". If not specified, the default vertical alignment is "middle". The meanings of the permitted values are the same as those described for the ALIGN attribute in subclause 14 on page 19.

80.4 Example

See clause 75 on page 55 for an example of the use of the <TR> element.

81 The TT element — Mono-spaced character style

81.1 Formal definition

```
<!ENTITY % physical.styles
           "... | TT | ..." >
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

81.2 Description

The contents of a <TT> element should be rendered in a mono-spaced font.

Both start and end tags are required.

81.3 Attributes

The <TT> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.

81.4 Example

```
The telex began:
<TT LANG=fr>CONFIDENTIEL</TT>
```

82 The U element — Underlined character style

82.1 Formal definition

```
<!ENTITY % physical.styles "... | U" >
<!ENTITY % dirn          "(ltr|rtl)" >
<!ELEMENT (%physical.styles;|...)  - -
          (%text;)+ >
<!ATTLIST (%physical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %physical.style.addon; >
```

82.2 Description

The contents of an <U> element should be rendered in an underlined text style if available, otherwise an alternative representation may be used.

Both start and end tags are required.

82.3 Attributes

The <U> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

82.4 Example

```
Browsers frequently
<U>underline</U>
the text associated with a link.
```

83 The UL element — Unordered list

83.1 Formal definition

```
<!ENTITY % dirn "(ltr|rtl)" >
<!ELEMENT (OL|UL)  - -  (LI)+ >
<!ATTLIST UL
          ALIGN   %justify; #IMPLIED
          CLASS   NAMES     #IMPLIED
          COMPACT (compact) #IMPLIED
          DIR     %dirn;    #IMPLIED
          ID      ID        #IMPLIED
          LANG    NAME      #IMPLIED
          %ul.addon; >
```

83.2 Description

The `<UL>` element represents a non empty unordered list of items usually rendered as a bulleted sequence.

The content is a sequence of `<LI>` elements [clause 55 on page 43].

Both start and end tags are required.

83.3 Attributes

The attributes of the `<UL>` element are as follows:

- ALIGN

  Please see subclause 9.1 on page 13.

- CLASS

  Please see subclause 9.2 on page 14.

- COMPACT

  The optional `COMPACT` attribute suggests that a compact rendering be used, perhaps because the list items are small, or the entire list is large. Unless the `COMPACT` attribute is present, a user agent may leave white space between successive `<LI>` elements.

  It is recommended that authors and authoring systems make use of the permitted omission of the attribute name, see ISO 8879:1986 subclause 7.9.1.2 [Gol90, p.329 and p.70 line 20]. This will facilitate the use of historic browsers.

- DIR

  Please see subclause 9.3 on page 14.

- ID

  Please see subclause 9.4 on page 14.

- LANG

  Please see subclause 9.5 on page 14.

83.4 Example

Note 67 Editors to NB experts: *Full details to be provided by Italian NB.*

```
<P LANG=it>Gelati
<UL>
<LI>Vanilla
<LI>Chocolate
<LI>Strawberry
</UL>
```

84 The VAR element — Generic variable

84.1 Formal definition

```
<!ENTITY % logical.styles "... | VAR" >
<!ENTITY % dirn           "(ltr|rtl)" >
<!ELEMENT (...|%logical.styles;)  - -
          (%text;)+ >
<!ATTLIST (%logical.styles;)
          CLASS NAMES  #IMPLIED
          DIR   %dirn; #IMPLIED
          ID    ID     #IMPLIED
          LANG  NAME   #IMPLIED
          %logical.style.addon; >
```

84.2 Description

The `<VAR>` element contains a placeholder variable, usually rendered in italic.

Both start and end tags are required.

84.3 Attributes

The <VAR> element has the following attributes.

- CLASS

  Please see subclause 9.2 on page 14.
- DIR

  Please see subclause 9.3 on page 14.
- ID

  Please see subclause 9.4 on page 14.
- LANG

  Please see subclause 9.5 on page 14.

84.4 Example

```
Type
<CODE>rm <VAR>file</VAR></CODE>
to delete
<VAR>file</VAR>
from directory.
```

85 Normative Annex 1 — Formal definitions

Note 68 *These formal definitions are not intended to be self documenting. The elements and attributes are described in the body of this Committee Draft. The SGML techniques are described in [Gol90].*

85.1 The SGML declaration

```
<!SGML  "ISO 8879:1986"

--

     ISO HyperText Markup Language Declaration,
     part of the ISO HyperText Markup Language (ISO-HTML) -
     an SGML system conforming to International Standard ISO 8879.

 (C) International Organization for Standardization 1997

     Permission to copy in any form is granted for use with
     conforming systems and applications as defined in ISO 15xxx,
     provided this copyright notice is included in all copies.

     This version based on:
       Id:     html.decl, v 1.17 1995/06/08
       Author: Daniel W. Connolly <connolly@w3.org>

--

CHARSET
      -- The following declaration provided by the IETF HTML
         internationalisation specification produces error
         messages with the nsgmls parser (SP-1.1.1):
         BASESET  "ISO Registration Number 176//CHARSET
                   ISO/IEC 10646-1:1993 UCS-2 with implementation level 3
                   //ESC 2/5 2/15 4/5"
         DESCSET    0     9 UNUSED
                    9     2      9
                   11     2 UNUSED
                   13     1     13
                   14    18 UNUSED
                   32    95     32
                  127     1 UNUSED
                  128    32 UNUSED
                  160 65374    160 --

      -- Use only 256 character positions until nsgmls problem solved --
         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
                ENTCAP          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 -- Deprecated --

         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
                  TAGLVL    100
                  TAGLEN   2100
                  GRPGTCNT  150
                  GRPCNT     64
                  ATTCNT     41  -- Allow Anders Berlund's ISO DTD --
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     ""
>
```

85.2 The entity set

```
<!-- iso-html.ent

     ISO HyperText Markup Language Entity Set,
     part of the ISO HyperText Markup Language (ISO-HTML) -
     an SGML system conforming to International Standard ISO 8879.

 (C) International Organization for Standardization 1997

     Permission to copy in any form is granted for use with
     conforming systems and applications as defined in ISO 15xxx,
     provided this copyright notice is included in all copies.

-->

               <!-- Interface for HTML extensions -->

<!-- Extensions may be pre-empted by placing earlier definitions of the
     following parameter entities in the document type declaration subset -->

<!ENTITY % head.extend   "NOP" -- Extend head content model -->
<!ENTITY % body.extend   "NOP" -- Extend body content model -->

<!ENTITY % a.addon          "" -- Extend a element attribute set -->
<!ENTITY % address.addon    "" -- Extend address element attribute set -->
<!ENTITY % area.addon       "" -- Extend area element attribute set -->
<!ENTITY % base.addon       "" -- Extend base element attribute set -->
<!ENTITY % bdo.addon        "" -- Extend bdo element attribute set -->
<!ENTITY % blockquote.addon "" -- Extend blockquote element attribute set -->
<!ENTITY % body.addon       "" -- Extend body element attribute set -->
<!ENTITY % br.addon         "" -- Extend br element attribute set -->
<!ENTITY % caption.addon    "" -- Extend caption element attribute set -->
<!ENTITY % dd.addon         "" -- Extend dd element attribute set -->
<!ENTITY % div.addon        "" -- Extend div element attribute set -->
<!ENTITY % dl.addon         "" -- Extend dl element attribute set -->
<!ENTITY % dt.addon         "" -- Extend dt element attribute set -->
<!ENTITY % form.addon       "" -- Extend form element attribute set -->
<!ENTITY % head.addon       "" -- Extend head element attribute set -->
```

```
<!ENTITY % hr.addon         "" -- Extend hr element attribute set -->
<!ENTITY % img.addon        "" -- Extend image element attribute set -->
<!ENTITY % input.addon      "" -- Extend input element attribute set -->
<!ENTITY % isindex.addon    "" -- Extend isindex element attribute set -->
<!ENTITY % iso-html.addon   "" -- Extend iso-html element attribute set -->
<!ENTITY % li.addon         "" -- Extend li element attribute set -->
<!ENTITY % link.addon       "" -- Extend link element attribute set -->
<!ENTITY % map.addon        "" -- Extend map element attribute set -->
<!ENTITY % meta.addon       "" -- Extend meta element attribute set -->
<!ENTITY % object.addon     "" -- Extend object element attribute set -->
<!ENTITY % ol.addon         "" -- Extend ol element attribute set -->
<!ENTITY % option.addon     "" -- Extend option element attribute set -->
<!ENTITY % p.addon          "" -- Extend p element attribute set -->
<!ENTITY % param.addon      "" -- Extend param element attribute set -->
<!ENTITY % pre.addon        "" -- Extend pre element attribute set -->
<!ENTITY % select.addon     "" -- Extend select element attribute set -->
<!ENTITY % span.addon       "" -- Extend span element attribute set -->
<!ENTITY % table.addon      "" -- Extend table element attribute set -->
<!ENTITY % td.addon         "" -- Extend td element attribute set -->
<!ENTITY % textarea.addon   "" -- Extend textarea element attribute set -->
<!ENTITY % th.addon         "" -- Extend th element attribute set -->
<!ENTITY % title.addon      "" -- Extend title element attribute set -->
<!ENTITY % tr.addon         "" -- Extend tr element attribute set -->
<!ENTITY % ul.addon         "" -- Extend ul element attribute set -->

<!ENTITY % heading.addon    "" -- Extend heading elements attribute set -->

<!ENTITY % physical.style.addon "" -- Extend physical character style
                                      attribute set -->
<!ENTITY % logical.style.addon  "" -- Extend logical character style
                                      attribute set -->
               <!-- Tokens defined by other standards -->

<!ENTITY % Content-Type "CDATA" -- MIME content type, RFC1521 -->
<!ENTITY % HTTP-Method "(get | post)" -- as per HTTP/1.1 RFC2068  -->
<!ENTITY % URL "CDATA" -- Uniform Resource Locator, RFC1808 RFC1738 -->

               <!-- Element tokens -->

<!ENTITY % special "A | BDO | BR | IMG | OBJECT | MAP | SPAN | %body.extend;" >

<!-- Logical character styles -->
<!ENTITY % logical.styles "CITE | CODE | DFN | EM | KBD | Q | SAMP |
                           STRONG | VAR" >

<!-- Physical character styles -->
<!ENTITY % physical.styles "B  | BIG | I | SMALL | STRIKE | SUB | SUP |
                            TT | U" >

               <!-- Model groups -->

<!-- Block-like elements eg. paragraphs and lists -->
<!ENTITY % block "BLOCKQUOTE | DIV | DL | FORM | HR | ISINDEX | OL |
```

```
                  P | PRE | TABLE | UL" >

<!-- Form fields - input elements that may appear only within forms -->
<!ENTITY % form.fields "INPUT | SELECT | TEXTAREA" >

<!-- Character level elements and text strings -->
<!ENTITY % text "#PCDATA | %physical.styles; | %logical.styles; | %special;" >
<!ENTITY % form.text "%text; | %form.fields;" >

<!-- Elements that may appear in a form, section or table -->
<!ENTITY % form.content    "(%block; | %form.text;)+" >
<!ENTITY % section.content "(%block; | %text; | ADDRESS)+" >
<!ENTITY % table.content   "(%block; | %text;)*" >

               <!-- Presentation dimensions and alignment -->

<!ENTITY % length "CDATA" -- nn for pixels or nn% for percentage length -->

<!-- Horizontal placement used by justifying elements -->
<!ENTITY % justify "(left|center|centre|right|justify)" >

<!-- Vertical alignment or placement -->
<!ENTITY % valign "(top|middle|bottom|baseline)" >
<!ENTITY % pixels "CDATA" -- integer representing length in pixels -->

               <!-- Presentation styles -->

<!ENTITY % shape     "(rect|circle|poly)" >
<!ENTITY % InputType "(checkbox | file | hidden | image | password |
                       radio | reset | submit | text)" >

<!-- Language dependent presentation - bidirectionalism -->

<!ENTITY % dirn "(ltr|rtl)" >

               <!-- Entities for special symbols -->

<!ENTITY quot  CDATA "&#34;"   -- double quote       -->
<!ENTITY amp   CDATA "&#38;"   -- ampersand          -->
<!ENTITY lt    CDATA "&#60;"   -- less than          -->
<!ENTITY gt    CDATA "&#62;"   -- greater than       -->
<!ENTITY nbsp  CDATA " "  -- non breaking space -->
<!ENTITY copy  CDATA "&#169;"  -- copyright sign     -->
<!ENTITY reg   CDATA "&#174;"  -- registered sign    -->

<!-- These entities require the full ISO 10646 based SGML declaration -->

<!--<!ENTITY zwnj  CDATA "&#8204;"-- -- Zero width non-joiner -->
<!--<!ENTITY zwj   CDATA "&#8205;"-- -- Zero width joiner     -->
<!--<!ENTITY lrm   CDATA "&#8206;"-- -- Left-to-right mark    -->
<!--<!ENTITY rlm   CDATA "&#8207;"-- -- Right-to-left mark    -->

<!-- SHORTREF mapping for the tab character -->
```

```
<!ENTITY   nontab  " " >
<!SHORTREF tabmap  "&#TAB;" nontab >
<!USEMAP   tabmap  ISO-HTML >

<!-- Use of the tab character is deprecated.  However, to facilitate
     the preparation of conforming documents by authors who use it,
     the tab character is allowed and is mapped into a single space. -->

<!-- End of file -->
```

85.3 The elements and attributes

```
<!-- iso-html.dtd

     ISO HyperText Markup Language Document Type Definition,
     an SGML system conforming to International Standard ISO 8879.

 (C) International Organization for Standardization 1997

     Permission to copy in any form is granted for use with
     conforming systems and applications as defined in ISO 15xxx,
     provided this copyright notice is included in all copies.

-->

<!ENTITY % version "CD" >

                 <!-- Document structure -->

<!--      ELEMENTS    MIN  CONTENT  (EXEPTIONS) -->
<!ELEMENT ISO-HTML    - -  (HEAD, BODY) >
<!ELEMENT HEAD        - -  (TITLE & ISINDEX? & BASE?)
                              +(LINK | META | %head.extend;) >
<!ELEMENT TITLE       - -  (#PCDATA)
                              -(LINK | META | %head.extend;) >
<!ELEMENT ISINDEX     - O  EMPTY>
<!ELEMENT BASE        - O  EMPTY >
<!ELEMENT LINK        - O  EMPTY >
<!ELEMENT META        - O  EMPTY >
<!ELEMENT BODY        - -  (%section.content;, (H1,B1)* ) >
<!ELEMENT H1          - -  (%text;)* >
<!ELEMENT B1          O O  (%section.content;, (H2,B2)* ) >
<!ELEMENT H2          - -  (%text;)* >
<!ELEMENT B2          O O  (%section.content;, (H3,B3)* ) >
<!ELEMENT H3          - -  (%text;)* >
<!ELEMENT B3          O O  (%section.content;, (H4,B4)* ) >
<!ELEMENT H4          - -  (%text;)* >
<!ELEMENT B4          O O  (%section.content;, (H5,B5)* ) >
```

```
<!ELEMENT H5          - -  (%text;)* >
<!ELEMENT B5          O O  (%section.content;, (H6,B6)* ) >
<!ELEMENT H6          - -  (%text;)* >
<!ELEMENT B6          O O  (%section.content;) >
<!ELEMENT ADDRESS     - -  (%text; | P)+ >
<!ELEMENT P           - O  (%text;)+ >
<!ELEMENT (OL|UL)     - -  (LI)+ >
<!ELEMENT LI          - O  %section.content; >
<!ELEMENT DL          - -  (DT|DD)+ >
<!ELEMENT DT          - O  (%text;)+ >
<!ELEMENT DD          - O  %section.content >
<!ELEMENT PRE         - -  (%text;)+ -(BIG|IMG|MAP|SMALL|SUB|SUP|
                                       %body.extend;) >
<!ELEMENT DIV         - -  %section.content; >
<!ELEMENT BLOCKQUOTE  - -  %section.content; >
<!ELEMENT FORM        - -  %form.content; -(FORM) >
<!ELEMENT INPUT       - O  EMPTY >
<!ELEMENT SELECT      - -  (OPTION)+ >
<!ELEMENT OPTION      - O  (#PCDATA) >
<!ELEMENT TEXTAREA    - -  (#PCDATA) >
<!ELEMENT HR          - O  EMPTY >
<!ELEMENT TABLE       - -  (CAPTION?, TR+) >
<!ELEMENT CAPTION     - -  (%text;)+ >
<!ELEMENT TR          - O  (TH|TD)+ >
<!ELEMENT (TH|TD)     - O  %table.content; >
<!ELEMENT (%physical.styles;|%logical.styles;)
                      - -  (%text;)+ >
<!ELEMENT A           - -  (%text;)* -(A) >
<!ELEMENT IMG         - O  EMPTY >
<!ELEMENT OBJECT      - -  (PARAM | %section.content;)* >
<!ELEMENT PARAM       - O  EMPTY >
<!ELEMENT BR          - O  EMPTY >
<!ELEMENT MAP         - -  (AREA)+ >
<!ELEMENT AREA        - O  EMPTY >
<!ELEMENT SPAN        - -  (%text;)+ >
<!ELEMENT BDO         - -  (%text;)+ >
<!ELEMENT NOP         - O  EMPTY --for DTD internal use only -->

              <!-- Attribute definition lists -->

<!--      ELEMENTS NAME       VALUE       DEFAULT -->
<!ATTLIST A        CHARSET    NAME       #IMPLIED -- MIME charset --
                   CLASS      NAMES      #IMPLIED -- element class --
                   DIR        %dirn;     #IMPLIED -- text direction --
                   HREF       %URL;      #IMPLIED -- source anchor --
                   ID         ID         #IMPLIED
                   LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                   NAME       CDATA      #IMPLIED -- target anchor --
                   REL        CDATA      #IMPLIED -- forward link types --
                   REV        CDATA      #IMPLIED -- reverse link types --
                   TITLE      CDATA      #IMPLIED -- advisory title str --
                   %a.addon; >
<!ATTLIST ADDRESS  ALIGN      %justify;  #IMPLIED -- text alignment --
```

```
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %address.addon; >
<!ATTLIST AREA    ALT        CDATA     #REQUIRED
                  COORDS     CDATA      #IMPLIED
                  -- comma separated list of numbers --
                  HREF       %URL;      #IMPLIED
                  NOHREF     (nohref)   #IMPLIED -- region has no action --
                  SHAPE      %shape;        rect
                  %area.addon; >
<!ATTLIST BASE    HREF       %URL      #REQUIRED
                  %base.addon; >
<!ATTLIST BDO     DIR        %dirn;    #REQUIRED
                  LANG       NAME       #IMPLIED
                  %bdo.addon; >
<!ATTLIST BLOCKQUOTE
                  ALIGN      %justify;  #IMPLIED -- text alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %blockquote.addon; >
<!ATTLIST BODY    CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %body.addon; >
<!ATTLIST BR      NOP        (nop)      #IMPLIED -- not for use in docs --
                  %br.addon; >
<!ATTLIST CAPTION ALIGN      (top|bottom)
                                        #IMPLIED
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %caption.addon; >
<!ATTLIST DD      CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %dd.addon; >
<!ATTLIST DIV     ALIGN      %justify;  #IMPLIED -- text alignment --
                  %div.addon; >
<!ATTLIST DL      CLASS      NAMES      #IMPLIED -- element class --
                  COMPACT    (compact)  #IMPLIED
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %dl.addon; >
<!ATTLIST DT      CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
```

```
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %dt.addon; >
<!ATTLIST FORM    ACTION     %URL;     #REQUIRED -- server-side handler --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ENCTYPE    %Content-Type;
                                       "application/x-www-form-urlencoded"
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  METHOD     %HTTP-Method;
                                             get -- see HTTP specification --
                  %form.addon; >
<!ATTLIST HR      ALIGN      %justify;  #IMPLIED
                  DIR        %dirn;     #IMPLIED -- text direction --
                  %hr.addon; >
<!ATTLIST HEAD    CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %head.addon; >
<!ATTLIST IMG     ALIGN      (top|middle|bottom|left|right)
                                        #IMPLIED -- wrt baseline --
                  ALT        CDATA     #REQUIRED -- in place of image --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  ISMAP      (ismap)    #IMPLIED -- use server image map --
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  SRC        %URL;     #REQUIRED -- URL of image to embed --
                  USEMAP     %URL;      #IMPLIED -- client-side image map --
                  %img.addon;                    -- additional attributes -->
<!ATTLIST INPUT   CHECKED    (checked)  #IMPLIED -- for radio, checkbox --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  MAXLENGTH  NUMBER     #IMPLIED
                  NAME       CDATA      #IMPLIED -- except submit, reset --
                  SIZE       CDATA      #IMPLIED -- specific to field type --
                  SRC        %URL;      #IMPLIED -- background image --
                  TYPE       %InputType;    text -- widget --
                  VALUE      CDATA      #IMPLIED -- for radio, checkbox --
                  %input.addon;                  -- additional attributes -->
<!ATTLIST ISINDEX CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  PROMPT     CDATA      #IMPLIED -- prompt message --
                  %isindex.addon; >
<!ATTLIST ISO-HTML CLASS     NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
```

```
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  VERSION    CDATA        #FIXED    %version;
                  %iso-html.addon; >
<!ATTLIST LI      ALIGN      %justify;  #IMPLIED -- text alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  VALUE      NUMBER     #IMPLIED -- reset sequence number --
                  %li.addon; >
<!ATTLIST LINK    CHARSET    NAME       #IMPLIED -- MIME charset --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  HREF       %URL;      #IMPLIED
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  REL        CDATA      #IMPLIED -- forward link types --
                  REV        CDATA      #IMPLIED -- reverse link types --
                  TITLE      CDATA      #IMPLIED -- advisory title str --
                  %link.addon; >
<!ATTLIST MAP     NAME       CDATA     #REQUIRED
                  %map.addon; >
<!ATTLIST META    CONTENT    CDATA     #REQUIRED -- associated information --
                  HTTP-EQUIV NAME       #IMPLIED
                  -- HTTP response header name --
                  NAME       NAME       #IMPLIED -- metainformation name --
                  %meta.addon; >
<!ATTLIST OBJECT  ALIGN      (texttop,middle,textmiddle,baseline,
                              textbottom,left,center,centre,right)
                                        #IMPLIED -- positioning in doc --
                  CLASS      NAMES      #IMPLIED -- element class --
                  CLASSID    %URL;      #IMPLIED -- identifies implement. --
                  CODEBASE   %URL;      #IMPLIED -- needed by some systems --
                  CODETYPE   CDATA      #IMPLIED -- code media type --
                  DATA       %URL;      #IMPLIED -- ref to object's data --
                  DECLARE    (declare)  #IMPLIED -- dont instantiate --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  NAME       %URL;      #IMPLIED -- submit with form --
                  STANDBY    CDATA      #IMPLIED -- show msg while loading --
                  TYPE       CDATA      #IMPLIED -- data media type --
                  USEMAP     %URL;      #IMPLIED -- ref to image map --
                  %object.addon; >
<!ATTLIST OL      ALIGN      %justify;  #IMPLIED -- text alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  COMPACT    (compact)  #IMPLIED
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  START      NUMBER     #IMPLIED -- start sequence number --
                  %ol.addon; >
<!ATTLIST OPTION  CLASS      NAMES      #IMPLIED -- element class --
```

```
                 DIR        %dirn;     #IMPLIED -- text direction --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 SELECTED   (selected) #IMPLIED
                 VALUE      CDATA      #IMPLIED -- defaults to content --
                 %option.addon; >
<!ATTLIST P      ALIGN      %justify;  #IMPLIED -- text alignment --
                 CLASS      NAMES      #IMPLIED -- element class --
                 DIR        %dirn;     #IMPLIED -- text direction --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 %p.addon; >
<!ATTLIST PARAM  NAME       NAME      #REQUIRED -- name of parameter --
                 TYPE       CDATA      #IMPLIED -- Internet Media Type --
                 VALUE      CDATA      #IMPLIED -- value of parameter --
                 VALUETYPE  (data|ref|object)
                                           data -- interpret value as --
                 %param.addon; >
<!ATTLIST PRE    CLASS      NAMES      #IMPLIED -- element class --
                 DIR        %dirn;     #IMPLIED -- text direction --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 WIDTH      NUMBER     #IMPLIED
                 %pre.addon; >
<!ATTLIST SELECT CLASS      NAMES      #IMPLIED -- element class --
                 DIR        %dirn;     #IMPLIED -- text direction --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 MULTIPLE   (multiple) #IMPLIED
                 NAME       CDATA     #REQUIRED
                 SIZE       NUMBER     #IMPLIED
                 %select.addon; >
<!ATTLIST SPAN   CLASS      NAMES      #IMPLIED -- element class --
                 DIR        %dirn;     #IMPLIED -- text direction --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 %span.addon; >
<!ATTLIST TABLE  ALIGN      (left,center,centre,right)
                                           left -- posn. rel. to window --
                 BORDER     %pixels;   #IMPLIED -- frame around table --
                 CELLPADDING %pixels;  #IMPLIED -- spacing within cells --
                 CELLSPACING %pixels;  #IMPLIED -- spacing between cells --
                 CLASS      NAMES      #IMPLIED -- element class --
                 DIR        %dirn;     #IMPLIED -- text direction --
                 DUMMY      (border)   #IMPLIED -- border w/o value --
                 ID         ID         #IMPLIED
                 LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                 WIDTH      %length;   #IMPLIED
                 -- table width in pixels or relative to window --
                 %table.addon; >
<!ATTLIST TD     ALIGN      %justify;  #IMPLIED -- horizontal alignment --
                 CLASS      NAMES      #IMPLIED -- element class --
                 COLSPAN    NUMBER            1 -- cols spanned by cell --
```

```
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  NOWRAP     (nowrap)   #IMPLIED -- suppress word wrap --
                  ROWSPAN    NUMBER            1 -- rows spanned by cell --
                  VALIGN     %valign;   #IMPLIED -- vertical alignment --
                  %td.addon; >
<!ATTLIST TEXTAREA CLASS     NAMES      #IMPLIED -- element class --
                  COLS       NUMBER    #REQUIRED
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  NAME       CDATA     #REQUIRED
                  ROWS       NUMBER    #REQUIRED
                  %textarea.addon; >
<!ATTLIST TH      ALIGN      %justify;  #IMPLIED -- horizontal alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  COLSPAN    NUMBER            1 -- columns spanned by cell --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  NOWRAP     (nowrap)   #IMPLIED -- suppress word wrap --
                  ROWSPAN    NUMBER            1 -- rows spanned by cell --
                  VALIGN     %valign;   #IMPLIED -- vertical alignment --
                  %th.addon; >
<!ATTLIST TITLE   CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %title.addon; >
<!ATTLIST TR      ALIGN      %justify;  #IMPLIED -- horizontal alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  VALIGN     %valign;   #IMPLIED -- vertical alignment --
                  %tr.addon; >
<!ATTLIST UL      ALIGN      %justify;  #IMPLIED -- text alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  COMPACT    (compact)  #IMPLIED
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %ul.addon; >

              <!-- Attribute group definition lists -->

<!ATTLIST (%physical.styles;)
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag--
                  %physical.style.addon; >
```

```
<!ATTLIST (%logical.styles;)
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %logical.style.addon; >

<!ATTLIST (H1 | H2 | H3 | H4 | H5 | H6)
                  ALIGN      %justify;  #IMPLIED -- text alignment --
                  CLASS      NAMES      #IMPLIED -- element class --
                  DIR        %dirn;     #IMPLIED -- text direction --
                  ID         ID         #IMPLIED
                  LANG       NAME       #IMPLIED -- RFC 1766 language tag --
                  %heading.addon; >

<!-- End of file -->
```

86 Informative Annex 2 — Sample marked up documents

Note 69 Editors to NB experts: *Sample documents will not be provided until any corrections resulting from the CD balloting process have been folded into the text.*

87 Informative Annex 3 — Bibliography

References

[Alv95] H. Alvestrand. *RFC1766 Tags for the Identification of Languages.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1766.txt>., March 1995.

[BF93] N. Borenstein and N. Freed. *RFC1521 MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for specifying and describing the format of Internet message bodies.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1521.txt>., September 1993.

[BL94] Tim Berners-Lee. *RFC1630 Uniform 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.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1630.txt>., June 1994.

[BLC95] Tim Berners-Lee and Daniel W. Connolly. *RFC1866 HyperText Markup Language - 2.0.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1866.txt>., September 22nd 1995. 'Proposed Standard'.

[BLMM94] Tim Berners-Lee, Larry Masinter, and M. McCahill. *RFC1738 Uniform Resource Locators (URL).* Internet Engineering Task Force asasasasasasasas <URL:ftp://ds.internic.net/rfc/rfc1738.txt>., December 1994.

[FGM+97] Roy T. Fielding, Jim Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, and Tim Berners-Lee. *RFC2068 HyperText Transfer Protocol — HTTP/1.1.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc2068.txt>., January 1997.

[Fie95] Roy Fielding. *RFC1808 Relative Uniform Resource Locators.* Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1808.txt>., June 1995.

[Gol90] Charles F. Goldfarb. *The SGML Handbook.* Clarendon Press, Oxford University Press, Walton Street, Oxford, first edition, 1990. Use in preference to the ISO 8879 document which it contains.

[Gri92] Barbara F. Grimes. *Ethnologue, Languages of the World.* Summer Institute of Linguistics, Dallas, 12th edition, 1992. Normative extension of IETF RFC 1766.

[HyT92] *ISO/IEC IS 10744, Hypermedia/Time-based Structuring Language (HyTime)*. ISO, Geneva, first edition, August 1992.

[KMS+97] Charlie Kindel, Lou Montulli, Eric Sink, Wayne Gramlich, Jonathan Hirschman, and Tim Berners-Lee. Inserting objects into HTML — work in progress. Technical Report WD-object-960422, World Wide Web Consortium, 1997. This working draft extends W3C Recommendation HTML 3.2 to support the insertion of multimedia objects.

[NM95] E. Nebel and L. Masinter. *RFC1867 Form-based File Upload in HTML*. Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1867.txt>., November 1995.

[Rag96] Dave Raggett. *HTML Tables*. Internet Engineering Task Force <URL:ftp://ds.internic.net/internet-drafts/draft-ietf-html-tables-05.txt>., January 23rd 1996. Work in progress.

[Rag97] Dave Ragget. HTML 3.2 Reference Specification. Technical Report REC-html32, World Wide Web Consortium, 1997. This Recommendation aims to capture recommended practice as of early 1996.

[RP94] J. Reynolds and J. Postel. *RFC1700 Assigned Numbers*. Internet Engineering Task Force <URL:ftp://ds.internic.net/rfc/rfc1700.txt>., October 1994.

[SGM86] *ISO/IEC IS 8879, Standard Generalized Markup Language (SGML)*. ISO, Geneva, first edition, October 1986.

[YNAD95] François Yergeau, Gavin T. Nicol, Glenn Adams, and Martin J. Dürst. *Internationalisation of the HyperText Markup Language*. Internet Engineering Task Force <URL:ftp://ds.internic.net/internet-drafts/draft-ietf-html-i18n-02.txt>., November 22nd 1995. Work in progress.

End of document.