भारतीय मानक IS/ISO/IEC 23736-3 : 2020

Indian Standard

सूचना प्रोद्योगिकी — डिजीटल प्रकाशन — इपीयूबी 3.0.1

भाग 3 अंतर्वस्तु दस्तावेज

Information Technology — Digital Publishing — EPUB 3.0.1

Part 3 Content Documents

ICS 35.240.30

भारतीय मानक ब्यूरो
BUREAU OF INDIAN STANDARDS
मानक भवन, 9 बहादुर शाह ज़फर मार्ग, नई दिल्ली - 110002
MANAK BHAVAN, 9 BAHADUR SHAH ZAFAR MARG
NEW DELHI - 110002
www.bis.gov.in www.standardsbis.in

May 2023 **Price Group 15**

Data Management System Sectional Committee, LITD 15

NATIONAL FOREWORD

This Indian Standard (Part 3) which is identical with ISO/IEC 23736-3 : 2020 ‘Information technology — Digital publishing — EPUB 3.0.1 — Part 3: Content documents’ issued by International Organization for Standardization (ISO) and International Electro technical Commission (IEC) jointly, was adopted by the Bureau of Indian Standards on the recommendations of the Data Management System Sectional Committee, LITD 15, and approval of the Electronics and Information Technology Division Council.

This Indian Standard is published in six parts. The other parts in this series are:

Part 1 Overview
Part 2 Publications
Part 4 Open container format
Part 5 Media overlays
Part 6 Canonical fragment identifiers

The text of ISO/IEC standard has been approved as suitable for publication as an Indian Standard without deviations. Certain conventions are, however, not identical to those used in Indian Standards. Attention is particularly drawn to the following:

a) Wherever the words ‘International Standard’ appear referring to this standard, they should be read as ‘Indian Standard’.
b) Comma (,) has been used as a decimal marker while in Indian Standards, the current practice is to use a point (.) as the decimal marker.

In this adopted standard, reference appears to certain International Standards for which the Indian Standard also exists. For undated references, the latest edition of the referenced document applies, including any corrigenda and amendment .The corresponding Indian Standard which is to be substituted in its respective place is listed below along with its degree of equivalence for the edition indicated:

International Standard	*Corresponding Indian Standard*	*Degree of Equivalence*
ISO/IEC 14496-22 : 2009 Information technology — Coding of audio-visual objects — Part 22: Open font format	IS/ISO/IEC 14496 : Part 22 : 2019 Information technology — Coding of audio-visual objects: Part 22 Open font format	Identical with ISO/IEC 14496-22 : 2019

The technical committee has reviewed the provisions of following International Standards referred in this adopted standard and has decided that they are acceptable for use in conjunction with this standard. For undated references, the latest edition of the referenced document applies, including any corrigenda and amendment:

International Standards	*Title*
RDFa 11	RDFa Core 1.1 — Second Edition. Syntax and processing rules for embedding RDF through attributes. Ben Adida, et al. 22 August 2013.
RFC 2046	Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types (RFC 2046) . N. Freed, N. Borenstein. November 1996.
RFC 2119	Key words for use in RFCs to Indicate Requirement Levels (RFC 2119). March 1997.
RFC 5646	Tags for Identifying Languages (RFC 5646). A. Phillips, M. Davis. September 2009.
WCAG 20	Web Content Accessibility Guidelines (WCAG) 2.0 . Ben Caldwel, et al. 11 December 2008.

Table of Contents

Indian Standard

INFORMATION TECHNOLOGY — DIGITAL PUBLISHING — EPUB 3.0.1
PART 3 CONTENT DOCUMENTS

1 Overview

1.1 Purpose and Scope

This section is informative

This specification, EPUB Content Documents 3.0.1, defines profiles of HTML5, SVG, and CSS for use in the context of EPUB® Publications.

This specification is one of a family of related specifications that compose EPUB 3, the third major revision of an interchange and delivery format for digital publications based on XML and Web Standards. It is meant to be read and understood in concert with the other specifications that make up EPUB 3:

- The EPUB 3 Overview [EPUB3Overview], which provides an informative overview of EPUB and a roadmap to the rest of the EPUB 3 documents. The Overview should be read first.
- EPUB Publications 3.0.1 [Publications301], which defines the semantics and overarching conformance requirements for each Rendition of an EPUB Publication.
- EPUB Open Container Format (OCF) 3.0.1 [OCF301], which defines a file format and processing model for encapsulating a set of related resources into a single-file (ZIP) EPUB Container.
- EPUB Media Overlays 3.0.1 [MediaOverlays301], which defines a format and a processing model for synchronization of text and audio.

This specification supersedes EPUB Content Documents 3.0 [ContentDocs30]. Refer to [EPUB3Changes] for information on differences between this specification and its predecessor.

› 1.2 Relationship to Other Specifications

This section is informative

› 1.2.1 Relationship to HTML5

The XHTML document type defined by this specification is based on W3C [HTML5], and inherits all definitions of semantics, structure and processing behaviors from the HTML5 specification unless otherwise specified.

In addition, this specification defines a set of extensions to the W3C HTML5 document model that Authors may include in XHTML Content Documents.

This specification defines a simplified processing model that does not require Reading Systems to support scripting, HTML5 forms or the HTML5 DOM. EPUB Reading Systems conformant with this specification are only required to be able to process a conforming EPUB Content Document. As support for scripting and HTML5 forms are optional Reading System features, a conformant Reading System might not be a fully-conformant HTML5 User Agent (i.e., it might not implement the complete HTML5 processing model).

› 1.2.2 Relationship to SVG

This specification defines a restricted subset of SVG 1.1 to represent vector graphics inline in XHTML Content Documents and as standalone SVG Content Documents.

› 1.2.3 Relationship to CSS

The CSS profile defined in this specification has CSS 2.1 [CSS2.1] as its baseline. Any CSS Style Sheet that conforms to CSS 2.1 may be used in the context of an EPUB Publication, except as noted in CSS 2.1.

This specification also incorporates features defined by CSS3 Modules and introduces EPUB-specific CSS constructs.

› 1.2.4 Future Maintenance

This specification references W3C specifications that are not yet final, and incompatible changes to them may occur in the future that would cause EPUB 3 Content Documents that were previously conformant to no longer be conformant to the latest versions of the referenced specifications.

The IDPF anticipates revising this specification if and when such incompatible changes occur, updating the normative constraints defined herein as necessary.

› 1.3 Terminology

EPUB Publication

A collection of one or more Renditions conforming to this specification and its sibling specifications , packaged in an EPUB Container.

An EPUB Publication typically represents a single intellectual or artistic work, but this specification and its sibling specifications do not circumscribe the nature of the content.

Rendition

A logical document entity consisting of a set of interrelated resources representing one rendering of an EPUB Publication.

Publication Resource

A resource that contains content or instructions that contribute to the logic and rendering of at least one Rendition of an EPUB Publication. In the absence of this resource, the EPUB Publication might not render as intended by the Author. Examples of Publication Resources include a Rendition's Package Document, EPUB Content Document, EPUB Style Sheets, audio, video, images, embedded fonts and scripts.

With the exception of the Package Document itself, the Publication Resources required to render a Rendition are listed in that Rendition's manifest [Publications301] and bundled in the EPUB Container file (unless specified otherwise in Publication Resource Locations [Publications301]).

Examples of resources that are not Publication Resources include those identified by the Package Document link [Publications301] element and those identified in outbound hyperlinks that resolve outside the EPUB Container (e.g., referenced from an [HTML5] `a` element `href` attribute).

Core Media Type Resource

A Publication Resource that is a Core Media Type and may therefore be included in the EPUB Publication without the provision of fallbacks [Publications301] .

EPUB Content Document

A Publication Resource that conforms to one of the EPUB Content Document definitions (XHTML or SVG).

An EPUB Content Document is a Core Media Type, and may therefore be included in the EPUB Publication without the provision of fallbacks [Publications301] .

XHTML Content Document

An EPUB Content Document conforming to the profile of [HTML5] defined in XHTML Content Documents.

XHTML Content Documents use the XHTML syntax of [HTML5].

SVG Content Document

An EPUB Content Document conforming to the constraints expressed in SVG Content Documents.

EPUB Navigation Document

A specialization of the XHTML Content Document, containing human- and machine-readable global navigation information, conforming to the constraints expressed in EPUB Navigation Documents.

Scripted Content Document

An EPUB Content Document that includes scripting or an XHTML Content Document that contains HTML5 forms elements.

Refer to Scripted Content Documents for more information.

Top-level Content Document

An EPUB Content Document referenced from the spine, whether directly or via a fallback chain [Publications301] .

Fixed-Layout Document

An EPUB Content Document directly referenced from the spine that has been designated `pre-paginated` in the Package Document, as defined in The rendition:layout Property [Publications301] .

The dimensions to use for rendering Fixed-Layout Documents are defined in Fixed-Layout Documents [ContentDocs301] .

Core Media Type

A set of Publication Resource types for which no fallback is required. Refer to Publication Resources [Publications301] for more information.

Package Document

A Publication Resource carrying bibliographical and structural metadata about a given Rendition of an EPUB Publication, as defined in Package Documents [Publications301] .

Manifest

A list of all Publication Resources that constitute the given Rendition of a EPUB Publication.

Refer to manifest [Publications301] for more information.

Spine

An ordered list of Publication Resources, typically EPUB Content Documents, representing the default reading order of the given Rendition of an EPUB Publication.

Refer to spine [Publications301] for more information.

Text-to-Speech (TTS)

The rendering of the textual content of an EPUB Publication as artificial human speech using a synthesized voice.

EPUB Style Sheet (or Style Sheet)

A CSS Style Sheet conforming to the CSS profile defined in *EPUB Style Sheets* .

Viewport

The region of an EPUB Reading System in which the content of an EPUB Publication is rendered visually to a User.

CSS Viewport

A Viewport capable of displaying CSS-styled content.

SVG Viewport

A Viewport capable of displaying SVG images.

EPUB Container (or Container)

The ZIP-based packaging and distribution format for EPUB Publications defined in [OCF301].

Author

The person(s) or organization responsible for the creation of an EPUB Publication, which is not necessarily the creator of the content and resources it contains.

User

An individual that consumes an EPUB Publication using an EPUB Reading System.

EPUB Reading System (or Reading System)

A system that processes EPUB Publications for presentation to a User in a manner conformant with this specification and its sibling specifications .

› 1.4 Typographic Conventions

The following typographic conventions are used in this specification:

`markup`

All markup (elements, attributes, properties), code (JavaScript, pseudo-code), machine processable values (string, characters, media types) and file names are in red-orange monospace font.

`markup`

Links to markup and code definitions are underlined and in red-orange monospace font. Only the first instance in each section is linked.

`http://www.idpf.org/`

URIs are in navy blue monospace font.

hyperlink

Hyperlinks are underlined and in blue.

[reference]

Normative and informative references are enclosed in square brackets.

Term

Terms defined in the Terminology are in capital case.

Term

Links to term definitions have a dotted blue underline. Only the first instance in each section is linked.

Normative element, attribute and property definitions are in blue boxes.

```
Informative markup examples are in white boxes.
```

NOTE

Informative notes are in yellow boxes with a "Note" header.

CAUTION

Informative cautionary note are in red boxes with a "Caution" header.

1.5 Conformance Statements

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in [RFC2119].

All sections of this specification are normative except where identified by the informative status label "This section is informative". The application of informative status to sections and appendices applies to all child content and subsections they may contain.

All examples in this specification are informative.

1.6 Namespace prefix mappings

For convenience, the following namespace prefix mappings [XMLNS] are used throughout this specification:

prefix	namespace URI
epub	http://www.idpf.org/2007/ops

m	http://www.w3.org/1998/Math/MathML
pls	http://www.w3.org/2005/01/pronunciation-lexicon
ssml	http://www.w3.org/2001/10/synthesis
svg	http://www.w3.org/2000/svg

2 EPUB Content Documents

2.1 XHTML Content Documents

This section defines a profile of [HTML5] for creating XHTML Content Documents. An instance of an XML document that conforms to this profile is a Core Media Type and is referred to in this specification and its sibling specifications as an XHTML Content Document.

Unless otherwise specified, this specification inherits all definitions of semantics, structure and processing behaviors from the [HTML5] specification.

CAUTION

The EPUB 3 XHTML Content Document definition references features in the W3C [HTML5] specification that are still works in progress and may change in incompatible ways. When utilizing such features, authors should consider the inherent risks in terms of the potential impact on interoperability and document longevity.

2.1.1 Content Conformance

An XHTML Content Document MUST meet all of the following criteria:

Document Properties

- It MUST meet the conformance constraints for XML documents defined in XML Conformance [Publications301] .
- It MUST be an [HTML5] document that conforms to the XHTML syntax.
- For all document constructs used that are defined by [HTML5], it MUST conform to the conformance criteria defined for those constructs in that specification, unless explicitly overridden in HTML5 Deviations and Constraints.
- It MAY include extensions to the [HTML5] grammar as defined in HTML5 Extensions, and MUST conform to all content conformance constraints defined therein.

File Properties

- The XHTML Content Document filename SHOULD use the file extension `.xhtml`.

NOTE
All Publication Resources referenced from an XHTML Content Document must conform to the constraints for Publication Resources defined in EPUB Publication — Content Conformance [Publications301]

› 2.1.2 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing XHTML Content Documents:

› Unless explicitly defined by this specification or its sibling specifications as overridden, it MUST process XHTML Content Documents using semantics defined by the [HTML5] specification and honor any applicable User Agent conformance constraints expressed therein.

› It MUST meet all Reading System conformance criteria defined in HTML5 Extensions.

› It MUST recognize and adapt behaviorally to the constraints defined in HTML5 Deviations and Constraints.

› It MUST meet the Reading System conformance criteria defined in Scripted Content Documents — Reading System Conformance.

› It MUST support visual rendering of XHTML Content Documents as defined in EPUB Style Sheets — Reading System Conformance.

› It SHOULD recognize embedded ARIA markup and support exposure of any given ARIA roles, states and properties to platform accessibility APIs [WAI-ARIA].

› 2.1.3 HTML5 Extensions

This section defines EPUB 3 XHTML Content Document extensions to the underlying [HTML5] document model.

› 2.1.3.1 Semantic Markup

› 2.1.3.1.1 Semantic Inflection

› 2.1.3.1.1.1 Introduction

This section is informative

Semantic inflection is the process of attaching additional meaning about the specific purpose and/or nature an element plays in an XHTML Content Document. In the context of EPUB Publications, the `epub:type` attribute is typically used to express domain-specific semantics, with the inflection(s) it carries complementing the underlying [HTML5] host vocabulary. The applied semantics always refine the meaning of their containing elements, never override their nature (e.g., the attribute can be used

to indicate a `section` is a chapter in a work, but cannot be used to turn `p` elements into list items to avoid proper list structures).

Semantic metadata is not intended for human consumption; it instead provides a controlled way for Reading Systems and other User Agents to learn more about the structure and content of a document, providing them the opportunity to enhance the reading experience for Users.

This specification defines a method for semantic inflection using *the attribute axis*: instead of adding new XML elements to the XHTML Content Document vocabulary, the `epub:type` attribute can be appended to existing elements to inflect the desired semantics. A mechanism to identify external vocabularies that provide controlled values for the attributes is also defined.

› 2.1.3.1.1.2 The `epub:type` Attribute

The `epub:type` attribute inflects semantics on the element on which it appears. Its value is one or more space-separated terms stemming from external vocabularies associated with the document instance, as defined in Vocabulary Association.

The inflected semantic MUST express a subclass of the semantic of the carrying element. In the case of semantically neutral elements (such as [HTML5] `div` and `span`), the inflected semantic MUST NOT attach a meaning that is already conveyed by an existing element (e.g., that a `div` represents a paragraph or section). Reading Systems MUST ignore inflected semantics that conflict with the carrying element.

As the [HTML5] `head` element is a metadata container for a document, structural semantics expressed on this element or any descendant of it have no meaning. Reading Systems MUST ignore such semantics.

NOTE

The `epub:type` attribute is intended to be functionally equivalent to the W3C Role Attribute [Role], but with restrictions as specified in Vocabulary Association. The IDPF's intent is to harmonize this attribute with W3C mechanisms for semantic inflection in a future major revision of the specification.

Attribute Name

`type`

Namespace

`http://www.idpf.org/2007/ops`

Usage

Global attribute. MAY be specified on all elements.

Value

A space-separated list of property [Publications301] values, with restrictions as defined in Vocabulary Association.

› *2.1.3.1.1.3 Vocabulary Association*

This specification adopts the vocabulary association mechanisms defined in Vocabulary Association Mechanisms [Publications301] , with the following modifications:

› Default Vocabulary

The default vocabulary for Content Documents is defined to be the EPUB 3 Structural Semantics Vocabulary.

› Reserved Prefixes

This specification reserves prefixes that Authors MAY use in the `epub:type` attribute in the normative document EPUB Content Documents Reserved Prefixes.

› The `prefix` Attribute

The `prefix` attribute definition is unchanged, but the attribute is defined to be in the namespace `http://www.idpf.org/2007/ops` when used in EPUB Content Documents.

The `prefix` attribute is only valid on the [HTML5] root `html` element.

› Examples

The following example shows the `epub:type` attribute used to inflect footnote and note reference semantics. The properties used are defined in the default vocabulary.

```
<html … xmlns:epub="http://www.idpf.org/2007/ops">
    …
    <p> … <a epub:type="noteref" href="#n1">1</a> … </p>
    …
    <aside epub:type="footnote" id="n1">
        …
    </aside>
    …
</html>
```

The following example shows the `epub:type` attribute used to inflect glossary semantics on an HTML5 definition list. The property used is defined in the default vocabulary.

```
<html … xmlns:epub="http://www.idpf.org/2007/ops">
    …
    <dl epub:type="glossary">
        …
    </dl>
    …
</html>
```

The following example shows the `epub:type` attribute used to inflect source publication pagebreak semantics. The property used is defined in the default vocabulary. (Note that the dc:source [Publications301] element provides a means of identifying the source publication to which the given pagination information applies.)

```
<html … xmlns:epub="http://www.idpf.org/2007/ops">
   …
  <p> … <span epub:type="pagebreak" title="234"/> … </p>
   …
</html>
```

› 2.1.3.1.1.4 Processing Requirements

A Reading System MUST process the `epub:type` attribute as follows:

- › It MAY associate specialized behaviors with none, some or all of the terms defined in the default vocabulary.
- › It MAY associate specialized behaviors with terms given in vocabularies other than the default.
- › It MUST ignore terms that it does not recognize.

When Reading System behavior associated with a given `epub:type` value conflicts with behavior associated with the carrying element, the behavior associated with the element MUST be given precedence.

› 2.1.3.1.2 Semantic Enrichment

› 2.1.3.1.2.1 Introduction

This section is informative

Unlike semantic inflection, which is about refining the structures within your markup, semantic enrichment enables the layering of meaning into the content in order to facilitate machine processing.

The [Microdata] and [RDFa11] specifications both define sets of attributes that can be used in XHTML Content Documents to semantically enrich the content.

› 2.1.3.1.2.2 Content Conformance

A conformant XHTML Content Document MUST meet all of the following criteria:

- › It MUST allow the use of [Microdata] attributes as defined in that specification.
- › It MUST allow the use of [RDFa11] attributes as defined in [HTML+RDFa11].

› 2.1.3.1.2.3 Processing Requirements

Reading Systems MAY process [Microdata] and [RDFa11] attributes as defined in their respective specifications, but such support is OPTIONAL.

› 2.1.3.2 SSML Attributes

› 2.1.3.2.1 Overview

The W3C Speech Synthesis Markup Language [SSML] is a language used for assisting Text-to-Speech (TTS) engines in generating synthetic speech. Although SSML is designed as a standalone document type, it also defines semantics suitable for use within other host languages.

This specification recasts the SSML 1.1 `phoneme` element as two attributes — `ssml:ph` and `ssml:alphabet` — and makes them available within EPUB XHTML Content Documents.

Reading Systems with Text-to-Speech (TTS) capabilities SHOULD support the SSML Attributes as defined below.

> NOTE
>
> For more information on EPUB 3 features related to synthetic speech, refer to Text-to-speech [EPUB3Overview] .

› 2.1.3.2.2 The `ssml:ph` attribute

The `ssml:ph` attribute specifies a phonemic/phonetic pronunciation of the text represented by the element to which the attribute is attached.

Attribute Name

`ph`

Namespace

`http://www.w3.org/2001/10/synthesis`

Usage

Global attribute. May be specified on all elements with which a phonetic equivalent can logically be associated (e.g., elements that contain textual information).

Must not be specified on a descendant of an element that already carries this attribute.

Value

A phonemic/phonetic expression, syntactically valid with respect to the phonemic/phonetic alphabet being used.

This attribute inherits all the semantics of the SSML 1.1 `phoneme` element `ph` attribute, with the following addition:

› When the `ssml:ph` attribute appears on an element that has text node descendants, the corresponding document text to which the pronunciation applies is the string that results from concatenating the descendant text nodes, in document order. The specified phonetic

pronunciation MUST therefore logically match the element's textual data in its entirety (i.e., not just an isolated part of its content).

NOTE

Reading Systems that support the SSML Attributes and PLS Documents must honor the defined precedence rules for these two constructs.

› 2.1.3.2.3 The `ssml:alphabet` attribute

The `ssml:alphabet` attribute specifies which phonemic/phonetic pronunciation alphabet is used in the value of the `ssml:ph` attribute.

Attribute Name

`alphabet`

Namespace

`http://www.w3.org/2001/10/synthesis`

Usage

Global attribute. MAY be specified on any element.

Value

The name of the pronunciation alphabet used in the value of `ssml:ph` (inherited).

This attribute inherits all the semantics of the SSML 1.1 `phoneme` element `alphabet` attribute, with the following addition:

› The value of the `ssml:alphabet` attribute is inherited in the document tree. The pronunciation alphabet used in a given `ssml:ph` attribute value is determined by locating the first occurrence of the `ssml:alphabet` attribute starting with the element on which the `ssml:ph` attribute appears, followed by the nearest ancestor element.

Reading Systems that support the SSML Attributes feature of this specification SHOULD support the IPA alphabet [refIPA], as expressed by the value "`ipa`".

› 2.1.3.3 Content Switching

› 2.1.3.3.1 Introduction

This section is informative

The `switch` element provides a simple mechanism through which Authors can tailor the content displayed to Users, one that isn't dependent on the scripting capabilities of the EPUB Reading System.

Reading System developers may choose to support XML vocabularies and new HTML elements that are not valid in XHTML Content Documents. The `switch` mechanism encourages this type of development and experimentation, but at the same time provides Authors who wish to take advantage of it the security of knowing that their content will still display on any compliant Reading System (i.e., it maintains the baseline requirement that all XHTML Content Documents be valid if none of the specialized markup is supported).

Content switching is not just about encouraging future development, however; it can also be used to create EPUB Publications that maintain a level of compatibility with older Reading Systems unable to handle the new features of EPUB 3. For example, instances of MathML, now a native type, could be added using `switch` elements so that EPUB 2 Reading Systems could instead provide fallback images or text.

› 2.1.3.3.2 Definition

› 2.1.3.3.2.1 The `epub:switch` Element

The `switch` element allows an XML fragment to be conditionally inserted into the content model of an XHTML Content Document.

Element name

`switch`

Namespace

`http://www.idpf.org/2007/ops`

Usage

In Flow and Inline content. Repeatable.

Attributes

`id` [optional]

The ID [XML] of this element, which MUST be unique within the document scope.

Content Model

In this order: `case` [1 or more], `default` [exactly 1].

A Reading System MUST individually process each `switch` element in a document to determine whether it can render any of the child `case` elements (as determined by the value of their `required-namespace` attributes).

For each `switch` encountered, the Reading System SHOULD render the content of the first `case` it supports, but is free to select from any of the available options. If the Reading System does not support the markup contained in any of the child `case` elements, it MUST render the contents of the `default` element.

The [HTML5] `object` element SHOULD be used to embed custom (non-core) content types in XHTML Content Documents. Custom markup SHOULD be wrapped in a `switch` element only when the content

it represents is an integral part of the document and depends on the context of the document to be properly processed.

The `switch` element is not intended to replace intrinsic fallback mechanisms, such as the [MATHML] `alttext` attribute and the [SVG] `title` and `desc` elements. Authors SHOULD always consider including intrinsic fallbacks, even when including a `switch` for Reading Systems with no support for the host grammar (e.g., to ensure accessibility).

› Examples

An example of ChemML markup inserted using the `switch` element.

```
<epub:switch id="cmlSwitch">

   <epub:case required-namespace="http://www.xml-cml.org/schema">
      <cml xmlns="http://www.xml-cml.org/schema">
         <molecule id="sulfuric-acid">
            <formula id="f1" concise="H 2 S 1 O 4"/>
         </molecule>
      </cml>
   </epub:case>

   <epub:default>
      <p>H<sub>2</sub>SO<sub>4</sub></p>
   </epub:default>

</epub:switch>
```

An example of adding MathML markup for compliance with EPUB 2 Reading Systems.

```
<epub:switch id="mathmlSwitch">

   <epub:case required-namespace="http://www.w3.org/1998/Math/MathML">
      <math xmlns="http://www.w3.org/1998/Math/MathML">
         <mrow>
            <mn>2</mn>
            <mo> &#x2061;<!--INVISIBLE TIMES--></mo>
            <mi>x</mi>
         </mrow>
         <mrow>
            <mo>+</mo>
            <mi>y</mi>
            <mo>-</mo>
            <mi>z</mi>
         </mrow>
      </math>
   </epub:case>

   <epub:default>
      <p>2x + y - z</p>
   </epub:default>

</epub:switch>
```

› *2.1.3.3.2.2 The `epub:case` Element*

The `case` element contains an instance of markup from an XML vocabulary. The included markup MAY be natively supported in XHTML Content Documents (in the case of MathML and SVG), but such support is not a requirement.

Element name

`case`

Namespace

`http://www.idpf.org/2007/ops`

Usage

Required first child of `switch` . Repeatable.

Attributes

`id` [optional]

The ID [XML] of this element, which MUST be unique within the document scope.

`required-namespace` [required]

An extension identifier in URI form [RFC2046] that identifies the XML vocabulary or extension that the Reading System MUST support in order to process the content of the `case` element.

Content Model

An XML fragment conforming to the markup vocabulary identified in the `required-namespace` attribute.

Each `case` element MUST contain an alternate representation of the same content. To ensure the best rendering of their content, Authors SHOULD order `case` elements by to their optimal rendering format.

If the `case` element contains markup that is valid in an XHTML Content Document (e.g., MathML), that content MUST be valid at the point where the `switch` element has been inserted (i.e., its addition MUST NOT result in an invalid document).

Foreign markup in a `case` element MUST be well formed, but does not have to be valid at its point of insertion. Authors SHOULD ensure that any foreign markup matches the context in which it is used (e.g., a block element should not be included in a `switch` element inserted in an inline context).

NOTE

The IDPF maintains an informative registry of common extension identifiers for use in the `required-namespace` attribute at `http://www.idpf.org/epub/switch/` .

› 2.1.3.3.2.3 The `epub:default` Element

The `default` element provides markup that is valid in any XHTML Content Document for when a Reading System cannot render any of the `case` elements.

Element name

`default`

Namespace

`http://www.idpf.org/2007/ops`

Usage

Required last child of `epub:switch` .

Attributes

`id` [optional]

The ID [XML] of this element, which MUST be unique within the document scope.

Content Model

An [HTML5]-compliant markup fragment.

The `default` element acts as a fallback for the `switch` and MUST include a representation of the content that is valid in XHTML Content Documents.

The `default` element MUST NOT include content that would invalidate the document at the point where the `switch` has been inserted: XHTML Content Documents MUST be valid if all the `switch` elements are replaced by their child `default` elements.

› 2.1.3.3.3 Processing

EPUB Reading Systems MUST support the `switch` element.

This specification does not require a specific rendering approach for `switch` elements. A Reading System MAY choose to apply CSS styling to render each `switch`, for example, but MAY use any other approach as appropriate. All Reading Systems MUST present the content of only one `case` element or the `default` element per `switch` for rendering, however.

The `switch` element MUST be processed as though all of its children but one have the HTML5 `hidden` attribute set (i.e., apply the same processing rules and requirements outlined for that attribute to the content not to be rendered).

NOTE

As the content that may be rendered depends on the capabilities of the User's Reading System, linking can be guaranteed only to the `switch` element. Deep referencing into the `switch` element is not recommended.

NOTE The occurrence of `switch` elements in XHTML Content Document is indicated in the Package Document manifest through the `switch [Publications301]` property.

› 2.1.3.4 The `epub:trigger` Element

The `trigger` element enables the creation of markup-defined user interfaces for controlling multimedia objects, such as audio and video playback, in both scripted and non-scripted contexts.

Element name

`trigger`

Namespace

`http://www.idpf.org/2007/ops`

Usage

As a child of `head` and in Flow content. Repeatable.

Attributes

`id` [optional]

The ID [XML] of this element, which MUST be unique within the document scope.

`action` [required]

The action to perform for this event.

Allowed values: `show` | `hide` | `play` | `pause` | `resume` | `mute` | `unmute`

`ref` [required]

An IDREF [XML] that identifies the element that is the object of the action.

`ev:defaultAction` [optional]

The applicable event for this trigger, as defined in [XML Events].

`ev:event` [required]

The applicable event for this trigger, as defined in [XML Events].

`ev:observer` [required]

The source object for this trigger, as defined in [XML Events].

`ev:phase` [optional]

The applicable event for this trigger, as defined in [XML Events].

`ev:propagate` [optional]

The applicable event for this trigger, as defined in [XML Events].

Content Model

Empty.

The `trigger` element associates an `event` from a specified source object (`observer`) with a desired action to be performed with a specified target object (`ref`).

The semantics of the defined `action` values are:

- `show` – set the element's DOM `visibility` [CSS2.1] property to visible.
- `hide` – set the element's DOM `visibility` [CSS2.1] property to hidden.
- `play` – play the associated resource from the beginning.
- `pause` – pause playing.
- `resume` – resume playing.
- `mute` – mute sound.
- `unmute` – unmute sound.

Reading Systems that support the [HTML5] `audio` or `video` elements MUST support the `epub:trigger` element. The `play`, `pause`, `resume`, `mute` and `unmute` actions can be applied to `audio` and `video` elements only. The `show` and `hide` actions can be applied to any descendant of the `body` element.

Sample markup of a video player that uses `trigger` elements to control playback and muting. The `role`, `tabindex` and `aria-controls` attributes ensure the `span` elements are accessible as buttons to keyboard users.

```
<html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:epub="http://www.idpf.org/2007/ops"
  xmlns:ev="http://www.w3.org/2001/xml-events">
  <head>
    <epub:trigger ev:observer="pause" ev:event="click" action="pause"
ref="test"/>
    <epub:trigger ev:observer="resume" ev:event="click" action="resume"
ref="test"/>
    <epub:trigger ev:observer="mute" ev:event="click" action="mute"
ref="test"/>
    <epub:trigger ev:observer="mute" ev:event="click" action="show"
ref="muted"/>
    <epub:trigger ev:observer="unmute" ev:event="click" action="unmute"
ref="test"/>
    <epub:trigger ev:observer="unmute" ev:event="click" action="hide"
ref="muted"/>
  </head>
  <body>
    <video id="test" src="birds.mp4" width="320" height="240"/>
    <p>
      <span id="resume" role="button" tabindex="0" aria-
controls="test">Play/Resume</span>
      <span id="pause" role="button" tabindex="0" aria-
controls="test">Pause</span>
      <span id="mute" role="button" tabindex="0" aria-
controls="test">Mute</span>
      <span id="unmute" role="button" tabindex="0" aria-
controls="test">Unmute</span>
```

```
      <span id="muted" role="button" tabindex="0" aria-
controls="test">MUTED</span>
    </p>
  </body>
</html>
```

› 2.1.3.5 Alternate Style Tags

In accordance with [AltStyleTags] , the `link` element `class` attribute MAY include any of the following values: `horizontal`, `vertical`, `day` and `night`. These values inherit the semantics defined by that specification for their use.

Reading Systems SHOULD select and utilize such tagged style sets as appropriate, and as described in that specification.

› 2.1.3.6 Custom Attributes

EPUB Reading Systems MAY introduce functionality not defined in this specification to enhance the rendering of EPUB Publications. To facilitate this experimentation, vendors MAY define custom attributes for use in XHTML Content Documents.

Custom attributes MAY be included on any element in an XHTML Content Document provided such attributes are from a foreign namespace, which is defined as a namespace [XMLNS] that does not map to either of the following URIs:

- `http://www.w3.org/1999/xhtml`
- `http://www.idpf.org/2007/ops`

Custom attributes, and the behaviors associated with them, MUST NOT alter the integrity of an EPUB Publication. The content MUST remain consumable by a User without any information loss or other significant deterioration, regardless of the Reading System it is rendered on.

> *NOTE*
>
> To facilitate interoperability of custom attributes across Reading Systems, vendors are strongly encouraged to document any extensions they implement at `http://www.idpf.org/epub/extensions/attributes` .

› 2.1.3.7 The `aria-describedat` Attribute

> *CAUTION*
>
> The `aria-describedat` attribute has been removed from ARIA 1.1. Use of the attribute in EPUB is now deprecated. Please see the errata for more information.

The `aria-describedat` attribute from [WAI-ARIA-1.1] MAY be specified on all elements in XHTML Content Documents, using the syntax and semantics defined in that specification. This attribute MAY

be used to reference descriptions outside the EPUB Container (see Publication Resource Locations [Publications301]).

Reading System support for this attribute is OPTIONAL.

NOTE

EPUB 3 does not support the full ARIA 1.1 specification at this time.

2.1.4 HTML5 Deviations and Constraints

This section defines deviations from, and constraints on, the underlying [HTML5] document model applicable to EPUB 3 XHTML Content Documents.

2.1.4.1 Embedded MathML

2.1.4.1.1 Introduction

This section is informative

XHTML Content Documents support embedded [MATHML] but limit its usage to a restricted subset of the full MathML markup language.

This subset is designed to ease the implementation burden on Reading Systems and to promote accessibility, while retaining compatibility with [HTML5] User Agents.

NOTE

The `mathml` `[Publications301]` property of the manifest `item` element indicates that an XHTML Content Document contains embedded MathML.

2.1.4.1.2 Content Conformance

Any occurrence of MathML markup in XHTML Content Documents MUST conform to the constraints expressed in the MathML specification [MATHML], with the following additional restrictions:

Presentation MathML

- The `m:math` element MUST contain only Presentation MathML, with the exception of the `m:annotation-xml` element as defined below.

Content MathML

- Content MathML MAY be included within MathML markup in XHTML Content Documents, and, when present, MUST occur within an `m:annotation-xml` child element of an `m:semantics` element.

› When Content MathML is included as per the previous condition, the given `m:annotation-xml` element's `encoding` attribute MUST be set to either of the functionally-equivalent values `MathML-Content` or `application/mathml-content+xml`, and its `name` attribute MUST be set to `contentequiv`.

Deprecated MathML

› Elements and attributes marked as deprecated in [MATHML] MUST NOT be included within MathML markup in XHTML Content Documents.

XHTML Content Document fragments

› XHTML Content Document fragments MAY be included within MathML markup in XHTML Content Documents, and, when present, MUST occur within an `m:annotation-xml` child element of an `m:semantics` element.

› When an XHTML Content Document fragment is included as per the above paragraph, the given `m:annotation-xml` element's `encoding` attribute MUST be set to `application/xhtml+xml` and its `name` attribute MUST be set to `alternate-representation`.

› Any included XHTML Content Document fragments MUST NOT themselves contain MathML markup.

› Any included XHTML Content Document fragments MUST conform to the content model in which the ancestor `m:math` element occurs, such that if the `m:math` element is replaced by the given XHTML Content Document fragment the document remains valid.

Alternative Content

› Alternative content SHOULD be included, and, when present, MUST be represented as defined in Alternative Content.

› 2.1.4.1.3 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing MathML embedded in XHTML Content Documents:

› It MUST support processing of Presentation MathML, and MAY support processing of Content MathML, using semantics defined by the MathML 3.0 specification [MATHML].

› If it has a Viewport, it MUST support visual rendering of Presentation MathML.

› When producing alternative textual content for MathML markup, it SHOULD be able to dynamically generate such content from the given Presentation MathML, and if not, MUST give preference to XHTML Content Document fragments followed by the `alttext` attribute on the `m:math` element.

› It MUST regard the mathml [Publications301] property of the Package Document manifest `item` element as the authoritative definition of whether an XHTML Content Document includes embedded MathML.

› 2.1.4.1.4 Alternative Content

Reading Systems SHOULD be able to generate any necessary alternative textual rendering dynamically using the given Presentation MathML markup (e.g., as output to Text-to-Speech (TTS) engines). To support Reading Systems that are not so capable, alternative textual content SHOULD be included with each occurrence of the `m:math` element in XHTML Content Documents.

The `alttext` attribute on the `m:math` element SHOULD be used for this purpose primarily when shorter alternative text runs are sufficient. When more extensive alternative text is required, XHTML Content Document fragments SHOULD be used. (Note that Reading Systems query these two alternative text locations using a defined preference order.)

For Reading System forward compatibility purposes, fallback images MAY be provided using the `altimg` attribute on the `m:math` element. It is RECOMMENDED that the dimension and alignment attributes (`altimg-width`, `altimg-height` and `altimg-valign`) be used in conjunction with the `altimg` attribute.

NOTE

All referenced Publication Resources must conform to the constraints for Publication Resources defined in EPUB Publication — Content Conformance [Publications301] .

› 2.1.4.2 Embedded SVG

XHTML Content Documents support the embedding of SVG 1.1 document fragments *by reference* (embedding via reference, for example, from an `img` or `object` element) and *by inclusion* (embedding via direct inclusion of the `svg:svg` element in the XHTML Content Document) [SVG].

The content conformance constraints for SVG embedded in XHTML Content Documents are the same as defined for SVG Content Documents in Restrictions on SVG 1.1.

Reading Systems MUST process SVG embedded in XHTML Content Documents as defined in SVG Content Documents — Reading System Conformance.

NOTE

The `svg [Publications301]` property of the manifest `item` element indicates that an XHTML Content Document contains embedded SVG.

› 2.1.4.2.1 Embedded SVG and CSS

For the purposes of styling SVG embedded in XHTML Content Documents *by reference*, Reading Systems MUST NOT apply CSS style rules of the containing document to the referenced SVG document.

For the purposes of styling SVG embedded in XHTML Content Documents *by inclusion*, Reading Systems MUST apply applicable CSS rules of the containing document to the included SVG elements.

NOTE

SVG included *by reference* is processed as a separate document, and may include its own CSS style rules just like an SVG Content Document would. Note that this is consistent with situations where an [HTML5] `object` element references an external [HTML5] element.

› 2.1.4.3 Unicode Restrictions

This section lists restrictions on the Unicode character repertoire.

Private Use Characters and Embedded Fonts

Any included characters that map to a code point within one of the Private Use Area (PUA) ranges as defined in [Unicode] MUST occur within a string that is styled or attributed in a manner that includes a reference to an embedded font that contains an appropriate glyph for that code point.

› 2.1.4.4 Discouraged Constructs

The `rp` Element

› The [HTML5] `rp` element is intended to provide a fallback — an optional parenthesis display around `ruby` markup — for older version Reading Systems that do not recognize ruby markup. As EPUB 3 Reading Systems are ruby-aware, and can provide fallbacks, the use of `rp` elements in Content Documents is discouraged.

The `embed` Element

› Since the [HTML5] `embed` element does not provide intrinsic facilities to provide fallbacks for Reading Systems that do not support scripting, its use is discouraged when the resource referenced has scripting components. Authors SHOULD use the `object` element instead.

› 2.1.4.5 Special Considerations

› 2.1.4.5.1 The `body` element

It is assumed, in formatting, that the default rendering for the [HTML5] `body` element is consistent with the [CSS2.1] `page-break-before` property having been set to `always`, but this default MAY be overridden by an appropriate style sheet declaration.

› 2.2 EPUB Navigation Documents

› 2.2.1 Introduction

This section is informative

The EPUB Navigation Document is a required component [Publications301] of EPUB Publications. It provides the Author with a mechanism to include a human- and machine-readable global navigation layer in the EPUB Publication, thereby ensuring increased usability and accessibility for the User.

The EPUB Navigation Document is an adaptation of XHTML Content Document and is, by definition, a valid XHTML Content Document instance. All Content and Reading System conformance

requirements that apply to XHTML Content Documents also apply to the EPUB Navigation Document.

The navigation features of this adaptation are expressed through specializations of the [HTML5] `nav` element. Each `nav` element in an EPUB Navigation Document represents a data island — an embedded source of specialized information within the general markup — from which Reading Systems can retrieve navigational information. Unlike typical XML data islands, however, the information within the `nav` element remains human readable as an [HTML5] document.

To facilitate machine readability, the content model of `nav` elements in EPUB Navigation Documents is restricted relative to what is allowed in general XHTML Content Documents.

Note that the navigation document is not exclusively for machine processing. Formulating the document as an XHTML Content Document enables its reuse in the linear reading order of an EPUB Publication, avoiding the creation of additional tables of contents (i.e., it can also be added to the spine [Publications301]). The visual display of components defined in the navigation document can be controlled using the hidden attribute, which has no effect outside of spine rendering (i.e., hiding a component from rendering in the spine does not hide it from Reading System presentation in a custom control).

When designing a navigation document for such dual use, however, be aware that machine extraction of the content can result in loss of formatting control. Scripting, styling and other HTML formatting can be stripped by a Reading System as it generates a custom control, such as the table of contents, from the markup. If such formatting and functionality is used, then the Navigation Document also needs to be included in the linear reading order. Another design consideration is to to use progressive enhancement techniques for scripting and styling of the navigation document, such that that the content will retain its integrity when rendered in a non-browser context.

NOTE

The EPUB Navigation Document is identified in the Package Document manifest through the `nav` `[Publications301]` property.

NOTE

The EPUB Navigation Document supersedes the NCX document type as defined in [OPF2].

Information on how EPUB 3 Publications may include an NCX document for EPUB 2 Reading System forwards compatibility purposes is available in NCX Superseded [Publications301] .

› 2.2.2 Content Conformance

A conformant EPUB Navigation Document MUST meet all of the following criteria:

Document Properties

› It MUST conform to all content conformance constraints for XHTML Content Documents as defined in XHTML Content Documents — Content Conformance.

› It MUST conform to all content conformance constraints specific for EPUB Navigation Documents expressed in EPUB Navigation Document Definition.

› As a conforming XHTML Content Document, it MAY be included in a Rendition's spine.

› 2.2.3 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing EPUB Navigation Documents:

› When requested by a User, Reading Systems MUST provide access to the links and link labels in the `nav` elements of the EPUB Navigation Document in a fashion that allows the User to activate the links provided. When a link is activated, the Reading System MUST relocate the application's current reading position to the destination identified by that link.

› Reading Systems MUST honor the above requirement irrespective of whether the EPUB Navigation Document provided in a Rendition is part of the spine.

› 2.2.4 EPUB Navigation Document Definition

› 2.2.4.1 The `nav` Element: Restrictions

When a `nav` element carries the `epub:type` attribute in an EPUB Navigation Document, this specification restricts the content model of the element and its descendants as follows:

› Each `nav` element MAY contain an optional heading indicating the title of the navigation list. The heading MUST be an HTML5 heading content element.

› The optional heading MUST be followed by a single `ol` ordered list; no other elements are permitted as direct children of the `nav` element. This ordered list represents the primary level of content navigation.

› Each list item (`li`) of the ordered list represents a primary heading, structure or other point of interest within the EPUB Publication and MUST contain either a child `a` element or a child `span` element. The `a` element describes the target within the Content Document that the link points to. The `span` element serves as a heading for breaking down lists into distinct groups (for example, a large list of illustrations can be segmented into several lists, one for each chapter).

› Each child `a` or `span` element of a list item MAY contain any valid HTML5 phrasing content, but MUST provide a non zero-length text label after concatenation of all child content and application of whitespace normalization rules. Although non-textual descendant elements MAY be rendered directly to Users, text content included in `title` or `alt` attributes MUST be used when determining compliance with this requirement.

› If an `a` or `span` element contains instances of HTML5 embedded content that do not provide intrinsic text alternatives, it MUST also include a `title` attribute with an alternate text rendering of the link label.

› The relative IRI reference provided in the `href` attribute of the `a` element MUST resolve to an EPUB Content Document or fragment therein.

› The `a` element MAY optionally be followed by an `ol` ordered list representing a subsidiary content level below that heading (e.g., all the subsection headings of a section). The `span` element MUST be followed by an `ol` ordered list: it cannot be used in "leaf" `li` elements. Regardless of whether an `a` or `span` element precedes it, this sublist MUST adhere to all the content requirements defined in this section for constructing the primary navigation list, and

recursively (for each additional level of the EPUB Publication's hierarchy represented in this manner).

› The `ol` element represents an ordered list. In the context of this specification, the default display style of list items MUST be equivalent to CSS `list-style: none` (Reading Systems with no CSS support MUST NOT show list item numbering). Authors MAY specify alternative list item styles using CSS, but these would obviously be ignored by Reading Systems that do not support Cascading Style Sheets.

IDPF specifications MAY introduce further restrictions on the content model defined above for `nav` elements in the EPUB Navigation Document.

The following example shows a partial `lot` ("list of tables") `nav` element, with `span` elements used as link-less headings for grouping the sublists.

```
<nav epub:type="lot">
    <h2>List of tables, broken down into individual groups, one per
major section of the publication content</h2>
    <ol>
        <li><span>Tables in Chapter 1</span>
            <ol>
                <li><a href="chap1.xhtml#table-1.1">Table 1.1</a>
                </li>
                <li><a href="chap1.xhtml#table-1.2">Table 1.2</a></li>
            </ol>
        </li>
        <li><span>Tables in Chapter 2</span>
            <ol>
                <li><a href="chap2.xhtml#table-2.1">Table 2.1</a>
                </li>
                <li><a href="chap2.xhtml#table-2.2">Table 2.2</a></li>
                <li><a href="chap2.xhtml#table-2.3">Table 2.3</a></li>
            </ol>
        </li>
        …
        <li><span>Tables in Appendix</span>
            <ol>
                <li><a href="appendix.xhtml#table-a.1">Table A.1</a>
                </li>
                <li><a href="appendix.xhtml#table-a.2">Table B.2</a>
</li>
            </ol>
        </li>
    </ol>
</nav>
```

› 2.2.4.2 The `nav` Element: Types

The `nav` elements defined in an EPUB Navigation Document are distinguished semantically by the value of their `epub:type` attribute. By default, values of `epub:type` are drawn from the EPUB 3 Structural Semantics Vocabulary [StructureVocab], but values drawn from other vocabularies are also allowed. Refer to The `epub:type` Attribute for more information.

› 2.2.4.2.1 The `toc` `nav` Element

The `toc nav` element defines the primary navigational hierarchy of the EPUB Publication. It conceptually corresponds to a table of contents in a printed work (i.e., it provides navigation to the structural sections of the EPUB Publication).

For usability and accessibility reasons, Authors SHOULD provide a comprehensive table of contents: the `toc nav` SHOULD NOT exclude references based solely on their nesting depth within the document hierarchy, as is often the case in print works (particularly in reduced tables of contents).

In the case of Renditions that exclusively reference XHTML Content Documents from their spines, the `toc nav` will typically correspond to the aggregation of HTML5 outlines of those documents (excluding any subtrees that do not contribute to the primary outline).

The order of `li` elements contained within the `toc nav` element MUST match the order of the targeted elements within each targeted EPUB Content Document, and MUST also follow the order of Content Documents in the Rendition's spine.

The `toc nav` element MUST occur exactly once in EPUB Navigation Documents.

NOTE

The `toc nav` element corresponds to the `navMap` element in the superseded NCX [OPF2].

› 2.2.4.2.2 The `page-list` `nav` Element

The `page-list nav` element is a container for pagination information. It provides navigation to positions in the content that correspond to the locations of page boundaries present in a print source being represented by the EPUB Publication.

The `page-list nav` element is OPTIONAL in EPUB Navigation Documents and MUST NOT occur more than once.

The order of `li` elements contained within a `page-list nav` structure MUST match the order of the actual pages inside each targeted EPUB Content Document and MUST also follow the order of Content Documents in the Rendition's spine.

The `page-list nav` element SHOULD contain only a single `ol` descendant (i.e., it SHOULD be a flat list, not a nested structure of navigation items).

NOTE

The `page-list nav` element corresponds to the `pageList` element in the superseded NCX. [OPF2]

NOTE

The dc:source [Publications301] element provides a means of identifying the source publication to which the given pagination information applies.

› 2.2.4.2.3 The `landmarks` `nav` Element

The `landmarks nav` element identifies fundamental structural components of the EPUB Publication in order to enable Reading Systems to provide the User efficient access to them.

The structural semantics of each link target within the `landmarks nav` element is determined by the value of the epub:type attribute on the `a` element descendants. The `epub:type` attribute is REQUIRED on `a` element descendants of the `landmarks nav` element.

The `landmarks nav` element extends the suggested HTML context of terms from the EPUB Structural Semantics Vocabulary to include the `a` element.

The following example shows a `landmarks nav` element with structural semantics drawn from the EPUB Structural Semantics Vocabulary.

```
<nav epub:type="landmarks">
    <h2>Guide</h2>
    <ol>
        <li><a epub:type="toc" href="#toc">Table of Contents</a></li>
        <li><a epub:type="loi" href="content.html#loi">List of
Illustrations</a></li>
        <li><a epub:type="bodymatter"
href="content.html#bodymatter">Start of Content</a></li>
    </ol>
</nav>
```

The `landmarks nav` element is OPTIONAL in EPUB Navigation Documents and MUST NOT occur more than once.

NOTE

The `landmarks nav` element corresponds to the deprecated OPF `guide` element. Refer to guide [Publications301] for more information.

› 2.2.4.2.4 Other `nav` Elements

EPUB Navigation Documents optionally MAY include one or more `nav` elements in addition to the `toc`, `page-list` and `landmarks nav` elements defined above. Such additional `nav` elements SHOULD have an `epub:type` attribute to provide a machine-readable semantic, and MUST have a human-readable heading as their first child.

This specification imposes no restrictions on the semantics of such additional `nav` elements: they MAY be used to represent navigational semantics for any information domain, and they MAY contain link targets with homogeneous or heterogeneous semantics.

› 2.2.4.3 The `hidden` attribute

In some cases, Authors may wish to hide parts of the navigation data within the content flow (i.e., the Reading System's principal rendering of the spine contents). A typical example is the list of page breaks, which usually isn't rendered as part of the content flow but instead exposed to the User separately in a dedicated navigation user interface.

While the CSS `display` property can be used to control the visual rendering of EPUB Navigation Documents in Reading Systems with CSS Viewports, not all Reading Systems provide such an interface. To control rendering across all Reading Systems, authors MUST use the [HTML5] `hidden` attribute to indicate which (if any) portions of the navigation data are excluded from rendering in the content flow. The `hidden` attribute has no effect on how navigation data is rendered outside of the content flow (such as in dedicated navigation user interfaces provided by Reading Systems).

The following example shows a partial `page-list` `nav` element. The presence of the `hidden` attribute on the root indicates that the entire list is excluded from rendering in the content flow.

```
<nav epub:type="page-list" hidden="">
    <h2>Pagebreaks of the print version, third edition</h2>
    <ol>
        <li><a href="frontmatter.xhtml#pi">I</a></li>
        <li><a href="frontmatter.xhtml#pii">II</a></li> … <li><a
href="chap1.xhtml#p1">1</a></li>
        <li><a href="chap1.xhtml#p2">2</a></li> … </ol>
</nav>
```

The following example shows a partial `toc` `nav` element where the `hidden` attribute is used to limit content flow rendering to the two topmost hierarchical levels.

```
<nav epub:type="toc" id="toc">
  <h1>Table of contents</h1>
  <ol>
    <li>
      <a href="chap1.xhtml">Chapter 1</a>
      <ol>
        <li>
          <a href="chap1.xhtml#sec-1.1">Chapter 1.1</a>
          <ol hidden="">
            <li>
              <a href="chap1.xhtml#sec-1.1.1">Section 1.1.1</a>
            </li>
            <li>
              <a href="chap1.xhtml#sec-1.1.2">Section 1.1.2</a>
            </li>
          </ol>
        </li>
        <li>
          <a href="chap1.xhtml#sec-1.2">Chapter 1.2</a>
        </li>
      </ol>
    </li>
    <li>
      <a href="chap2.xhtml">Chapter 2</a>
    </li>
  </ol>
</nav>
```

› 2.3 SVG Content Documents

› 2.3.1 Introduction

This section is informative

The Scalable Vector Graphics (SVG) 1.1 (Second Edition) specification [SVG] defines a format for representing final-form vector graphics and text.

Although an EPUB Publication typically uses XHTML Content Documents as the top-level document type, the use of SVG Content Documents is also permitted. SVGs are typically only used in certain special circumstances, such as when final-form page images are the only suitable representation of the content (as may be the case, for example, in the context of manga or comic books).

This section defines a profile for [SVG] documents. An instance of an XML document that conforms to this profile is a Core Media Type and is referred to in this specification and its sibling specifications as an SVG Content Document.

NOTE

This section defines conformance requirements for SVG Content Documents. Refer to Embedded SVG for conformance requirements for SVG embedded in XHTML Content Documents.

› 2.3.2 Content Conformance

An SVG Content Document MUST meet all of the following criteria:

Document Properties

› It MUST meet the conformance constraints for XML documents defined in XML Conformance [Publications301] .

› It MUST be an SVG 1.1 document fragment valid to the constructs defined in that specification, and conform to all content conformance constraints expressed in Restrictions on SVG 1.1.

› It SHOULD adhere to the accessibility guidelines given in [SVG Access].

File Properties

› The SVG Content Document filename SHOULD use the file extension `.svg`.

NOTE

All Publication Resources referenced from an SVG Content Document must conform to the constraints for Publication Resources defined in EPUB Publication — Content Conformance [Publications301]

› 2.3.3 Restrictions on SVG 1.1

This specification restricts the content model of SVG Content Documents and SVG embedded in XHTML Content Documents as follows:

› The [SVG] Animation Elements and Animation event attributes MUST NOT occur.

› The [SVG] `svg:foreignObject` element MUST contain either [HTML5] flow content or exactly one [HTML5] `body` element. This content MUST represent a valid document fragment of the XHTML Content Document model defined in XHTML Content Documents — Content Conformance. The `svg:foreignObject` element's `requiredExtensions` attribute, if given, MUST be set to `http://www.idpf.org/2007/ops`.

› The [SVG] `svg:title` element MUST contain only valid XHTML Content Document Phrasing content.

› 2.3.4 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing SVG Content Documents and SVG embedded in XHTML Content Documents:

› It MUST support the language features of SVG that correspond to the feature string `http://www.w3.org/TR/SVG11/feature#SVG-dynamic` minus the `http://www.w3.org/TR/SVG11/feature#Animation` and `http://www.w3.org/TR/SVG11/feature#AnimationEventsAttribute` features (see Feature strings) [SVG].

› It MUST meet the Reading System conformance criteria defined in Scripted Content Documents — Reading System Conformance.

› If it has an SVG Viewport, it MUST support the visual rendering of SVG using CSS as defined in Section 6 of [SVG], and it SHOULD support all properties defined in Appendix N of that specification. In the case of embedded SVG, it MUST also conform to the constraints defined in Embedded SVG and CSS.

› It SHOULD support User selection and searching of text within SVG elements.

› It MUST recognize the value `http://www.idpf.org/2007/ops` of the `requiredExtensions` attribute when appearing on the `svg:switch` and `svg:foreignObject` elements as representing the occurrence of XHTML Content Document fragments.

› It MUST regard the svg [Publications301] property of the Package Document manifest `item` element as the authoritative definition of whether an EPUB XHTML Content Document includes embedded SVG.

› 2.3.5 Semantic Inflection

The synatx and semantics defined in XHTML Semantic Inflection are inherited for use of the `epub:type` and `epub:prefix` attributes in SVG Content Documents.

The use of the `epub:prefix` attribute is valid on the root `svg` element in SVG Content Documents. Prefixes used in embedded SVG must be declared on the [HTML5] root `html` element, as defined in XHTML Semantic Inflection.

2.4 Scripted Content Documents

EPUB Content Documents MAY contain scripting using the facilities defined for this in the respective underlying specifications ([HTML5] and [SVG]). When an EPUB Content Document contains scripting, it is referred to in this specification and its sibling specifications as a Scripted Content Document. This label also applies to XHTML Content Documents when they contain instances of HTML5 forms.

2.4.1 Scripting Contexts

This specification defines two contexts in which scripts MAY appear:

spine-level

An instance of the [HTML5] `script` element included in a Top-level Content Document.

container-constrained

An instance of the [HTML5] `script` element included in an EPUB Content Document that is embedded in a parent Content Document using one of the [HTML5] `object` , `iframe` or `embed` elements.

In both of the above-defined contexts, whether the JavaScript code is embedded directly in the `script` element or referenced via its `src` attribute makes no difference to the executing context.

Which context a script is used in determines the rights and restrictions that a Reading System may place on it. Refer to Content Conformance and Reading System Conformance for some specific requirements that must be adhered to (not all Reading Systems may provide the same scripting functionality).

Example

Consider the following example Package Document:

```
<package …>
    …
    <manifest>
        …
        <item id="chap01"
            href="scripted01.xhtml"
            media-type="application/xhtml+xml"
            properties="scripted"/>
        <item id="inset01"
            href="scripted02.xhtml"
            media-type="application/xhtml+xml"
            properties="scripted"/>
        <item id="slideshowjs"
            href="slideshow.js"
            media-type="text/javascript"/>
    </manifest>

    <spine …>
```

```
        <itemref idref="chap01"/>
        …
    </spine>
    …
</package>
```

and the following file `scripted01.xhtml`:

```
<html …>
    <head>
        …
        <script type="text/javascript">
            alert("Reading System name: " +
navigator.epubReadingSystem.name);
        </script>
    </head>
    <body>
        …
        <iframe src="scripted02.xhtml" … />
        …
    </body>
</html>
```

and the following file `scripted02.xhtml`:

```
<html …>
    <head>
        …
        <script type="text/javascript" href="slideshow.js"></script>
    </head>
    <body>
        …
    </body>
</html>
```

From these examples, it is true that:

- the code in the `script` element in the `head` in `scripted01.xhtml` is a spine-level script because the document is referenced from the spine;
- the code in the `script` element in `scripted02.xhtml` is a container-constrained script because the HTML file it occurs in is included in `scripted01.xhtml` via the `iframe` element.

› 2.4.2 Content Conformance

Container-constrained scripts

› A container-constrained script MUST NOT contain instructions for modifying the DOM of the parent Content Document or other contents in the EPUB Publication, and MUST NOT contain instructions for manipulating the size of its containing rectangle.

Spine-level scripts

› EPUB Content Documents that include spine-level scripting MUST utilize the *progressive enhancement technique*, which for the purposes of this specification has the following definition: when the document is rendered by a Reading System without scripting support or with scripting support disabled, the top-level document content MUST retain its integrity, remaining consumable by the User without any information loss or other significant deterioration.

Accessibility

› EPUB Content Documents that include scripting — using any inclusion model — SHOULD employ relevant accessibility techniques to ensure that the content remains consumable by all Users. [WAI-ARIA] [WCAG20]

Fallbacks

› EPUB Content Documents that include scripting — using any inclusion model — MAY provide fallbacks for such content, either by using intrinsic fallback mechanisms (such as those available for the [HTML5] `object` and `canvas` elements) or, when an intrinsic fallback is not applicable, by using a manifest-level [Publications301] fallback.

NOTE

The `scripted` `[Publications301]` property of the manifest `item` element indicates that an EPUB Content Document is a Scripted Content Document.

› 2.4.3 Reading System Conformance

EPUB Reading System support for scripting is OPTIONAL. A Reading System that supports scripting MUST meet the following criteria:

› It MUST support container-constrained scripting and MAY support spine-level scripting.

› It MAY render Scripted Content Documents as an interactive, scripted User Agent according to [HTML5].

› It MUST NOT allow a container-constrained script to modify the DOM of the parent Content Document or other contents in the EPUB Publication, and MUST NOT allow it to manipulate the size of its containing rectangle. (Note: Even if a script is not container-constrained, the Reading System MAY impose restrictions on modifications (see also the dom-manipulation feature).)

› It MAY place additional limitations on the capabilities provided to scripts during execution (e.g., limiting networking).

› It MUST implement the JavaScript `navigator` extension object `epubReadingSystem` defined in Appendix A, *JavaScript epubReadingSystem Object*. It also MUST support the `dom-manipulation` and `layout-change` features defined in Features in container-constrained scripting contexts.

› It MUST regard the `scripted` `[Publications301]` property of the Package Document manifest `item` element as the authoritative definition of whether an EPUB Content Document includes scripting.

A Reading System that does not support scripting MUST meet the following criteria:

- It MUST process fallbacks for scripted content as defined in Fallbacks for Scripted Content Documents.

NOTE

Reading Systems may render Scripted Content Documents in a manner that disables other EPUB capabilities and/or provides a different rendering and User experience (e.g., by disabling pagination).

Authors choosing to restrict the usage of scripting to the container-constrained model will ensure a more consistent User experience between scripted and non-scripted content (e.g., consistent pagination behavior).

Authors should use declarative techniques whenever practical to increase the interoperability, longevity and accessibility of their EPUB Publications, and avoid the inclusion of scripting whenever practical.

2.4.4 Security Considerations

This section is informative

All EPUB Authors and Reading System developers need to be aware of the security issues that arise when scripted content is executed by a Reading System. As the underlying scripting model employed by Reading Systems and browsers is the same, the same kinds of issues encountered in Web contexts must be taken into consideration.

Each Reading System should establish if the scripts in a particular document are to be trusted or not. It is recommended that all scripts be treated as untrusted (and potentially malicious), and that all vectors of attack be examined and protected against. In particular, the following should be considered:

- an attack against the runtime environment (e.g., stealing files from a User's hard drive);
- an attack against the Reading System itself (e.g., stealing a list of a User's books or causing unexpected behavior);
- an attack of one Content Document against another (e.g., stealing data that originated in a different document);
- an attack of an unencrypted script against an encrypted portion of a document (e.g., an injected malicious script extracting protected content);
- an attack against the local network (e.g., stealing data from a server behind a firewall).

The following recommendations are provided as a guide to handling untrusted scripts:

- Reading Systems should behave as if a unique domain were allocated to each Content Document, as browser-based security relies heavily on document URLs and domains. Adopting this approach will isolate documents from each other and from other Internet domains, thereby limiting access to external URLs, cookies, DOM storage, etc.

Reading Systems that enable scripting and network access should also consider including methods to notify the user that network activity is occurring and/or that allow them to disable it.

NOTE

In practice, Reading Systems may share domains across documents, but they still should maintain isolation between documents.

If parts of a document are encrypted and parts are not, or if different encryption keys are used for different parts of the document, a unique per-document domain might not provide sufficient protection.

- If a Reading System allows persistent data to be stored, that data should be treated as sensitive. Scripts may save persistent data through cookies and DOM storage, but Reading Systems may block such attempts. Reading Systems that do allow data to be stored must ensure that it is not made available to other unrelated documents (e.g., ones that could have been spoofed). In particular, checking for a matching document identifier (or similar metadata) is not a valid method to control access to persistent data.

  Reading Systems that allow local storage should also provide methods for Users to inspect, disable, or delete that data. The data should be destroyed if the corresponding EPUB Publication is deleted.

Note that compliance with these recommendations does not guarantee protection from the possible attacks listed above; developers must examine each potential vulnerability within the context of their Reading System.

› 2.4.5 Event Model Considerations

This section is informative

Reading Systems should follow the DOM Event model as per [HTML5] and pass UI events to the scripting environment before performing any default action associated with these events. Reading System implementers should ensure that scripts cannot disable critical functionality (such as navigation) to constrain the extent to which a potentially malicious script could impact their Reading Systems. As a result, although the scripting environment should be able to cancel the default action of any event, some events either might not be passed through or might not be cancelable.

Authors should take into account the wide variety of possible Reading System implementations when adding scripting functionality to their EPUB Publications (e.g., not all devices have physical keyboard, and in many cases a soft keyboard is only activated only for text input elements). Consequently, relying on keyboard events alone is not recommended; alternative ways to trigger the desired action should always be provided.

› 2.5 Fixed-Layout Documents

› 2.5.1 Introduction

This section is informative

This section defines rules for the expression and interpretation of dimensional properties of EPUB Content Documents marked as `pre-paginated` in the Package Document.

This specification does not define how the the initial containing block, will be placed within the Reading System content display area.

> NOTE
>
> Refer to Fixed-Layout Properties [Publications301] for information on how to designate that a Rendition, or its individual spine items, are to be rendered in a pre-paginated manner (i.e., with fixed width and height dimensions).

› 2.5.2 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing Fixed-Layout Documents:

- It SHOULD allocate the full content rendering area for the document.
- It MUST use the dimensions expressed in the `viewport` `meta` tag to render XHTML Content Documents, but MAY obtain these dimensions from the package rendition:viewport property [Publications301] .
- It MUST use the dimensions expressed in the `viewBox` attribute to render SVG Content Documents, but MAY obtain these dimensions from the package rendition:viewport property [Publications301] .
- It MUST use the dimensions expressed in the content in the case of discrepancies with the package rendition:viewport property [Publications301] .

› 2.5.3 Viewport Rendering

When rendering Fixed-Layout Documents, the default intent is that the content rendering area SHOULD occupy as much of the available application screen area as possible. Reading Systems SHOULD NOT inject additional content such as border, margins, headers or footers into the viewport or the application area surrounding the viewport.

> NOTE
>
> The exposure of Reading System control widgets to the User is implementation-specific and not included in the above behavioral expectations.

› 2.5.4 Content Dimensions for XHTML and SVG

Each EPUB Content Document referenced from a spine item that has the `pre-paginated` value set MUST contain the `viewport` (for XHTML) or `viewBox` (for SVG) dimension expressions as defined

below, regardless of whether the value is set via the global rendition:layout property [Publications301] for the Rendition or on a spine-level override [Publications301] .

For both XHTML and SVG Content Documents, the dimension expressions define the CSS initial containing block (ICB) expressed in CSS Pixels [CSS2.1].

› 2.5.4.1 Expressing ICB Dimensions in XHTML

For XHTML Content Documents, the ICB dimensions MUST be expressed in a `viewport meta` tag using the syntax defined in [MetaTags]. In this version of this specification, only the width and height expressions MUST be recognized by Reading Systems.

The following example shows a `viewport meta` tag.

```
<head>
   …
   <meta name="viewport" content="width=1200, height=600"/>
   …
</head>
```

Reading Systems MUST clip XHTML content to the ICB dimensions declared in the `viewport meta` tag; content positioned outside of the initial containing block will not be visible. When the ICB aspect ratio does not match the aspect ratio of the Reading System content display area, Reading Systems MAY position the ICB inside the area to accommodate the user interface; in other words, added letter-boxing space MAY appear on either side (or both) of the content.

› 2.5.4.2 Expressing ICB Dimensions in SVG

For SVG Content Documents, the ICB dimensions MUST be expressed using the `viewBox` attribute [SVG].

The following example shows a `viewBox` attribute declaration.

```
<svg xmlns="http://www.w3.org/2000/svg"
     version="1.1" width="100%" height="100%"
     viewBox="0 0 844 1200">
   …
</svg>
```

› 3 EPUB Style Sheets

This section defines a profile for Cascading Style Sheets (CSS) intended to be used for styling of XHTML Content Documents. An instance of a CSS Style Sheet that conforms to this profile is a Core Media Type and is referred to in this specification and its sibling specifications as an EPUB Style Sheet.

CAUTION

The EPUB 3 CSS Profile references CSS specifications that are still works in progress and may change in incompatible ways. When utilizing features from such specifications, authors should consider the inherent risks in terms of the potential impact on interoperability and document longevity.

NOTE

The EPUB 3 CSS Profile employs the usage of the `-epub-` prefix for a number of CSS3 property names, as detailed below. As the CSS3 modules that define these properties mature and stabilize, EPUB authoring guidelines may encourage authors to also include unprefixed equivalents of these properties in EPUB 3 Style Sheets.

› 3.1 Content Conformance

A conformant EPUB Style Sheet MUST meet all of the following criteria:

- It MUST adhere to all content restrictions given in EPUB 3 CSS Profile.
- It MAY include constructs not explicitly identified in the EPUB 3 CSS Profile, but SHOULD be authored so that rendering fidelity does not depend on such additional constructs.
- It MUST be UTF-8 or UTF-16 encoded.

NOTE

All Publication Resources referenced from a CSS Style Sheet must conform to the constraints for Publication Resources defined in EPUB Publication — Content Conformance [Publications301]

› 3.2 Reading System Conformance

- Reading Systems with a CSS Viewport SHOULD support — render as defined by the corresponding specification in the Viewport — all CSS constructs included in this profile unless detailed otherwise in EPUB 3 CSS Profile.
- Reading Systems MAY support additional CSS constructs not explicitly identified in the EPUB 3 CSS Profile, and MUST handle any unsupported constructs as defined in [CSS2.1].

NOTE

Reading Systems have varying capabilities with regards to CSS rendering support, so may ignore some or all style information of an EPUB Style Sheet.

In addition, even when a Reading System does have a CSS Viewport, it is likely to render content in a manner that differs from typical HTML5 User Agents (e.g., paginating content rather than providing a infinitely scrolling surface).

› 3.3 EPUB 3 CSS Profile

› 3.3.1 CSS 2.1

The style baseline of the EPUB 3 CSS Profile is Cascading Style Sheets Level 2 Revision 1 [CSS2.1]. The profile includes all style sheet constructs normatively defined in [CSS2.1], with the following exceptions:

- The `absolute` value of the `position` property SHOULD be used only with XHTML Content Documents whose rendition:layout property [Publications301] has been set to `pre-paginated`.
- The `fixed` value of the `position` property is not part of the EPUB 3 CSS Profile. To avoid potential rendering and interoperability issues, it SHOULD NOT be included in an EPUB Style Sheet.
- The `direction` and `unicode-bidi` properties MUST NOT be included in an EPUB Style Sheet. Authors SHOULD use appropriate [HTML5] markup to express directionality information instead.

Reading Systems that have a CSS Viewport MUST support the `font-family` property.

› 3.3.2 CSS 2.0

The EPUB 3 CSS Profile includes the following values for the `list-style-type` property as defined in [CSS2.0]:

- `cjk-ideographic`
- `hebrew`
- `hiragana`
- `hiragana-iroha`
- `katakana`
- `katakana-iroha`

› 3.3.3 CSS 3.0 Speech

The EPUB 3 CSS Profile includes `-epub-` prefixed versions of the following properties from the CSS3 Speech Module [CSS3Speech] using syntax as defined in [CSS3Speech-20110818] and semantics as defined in [CSS3Speech]:

- `-epub-cue`
- `-epub-pause`
- `-epub-rest`

- `-epub-speak`
- `-epub-speak-as`
- `-epub-voice-family`

NOTE

For more information on EPUB 3 features related to synthetic speech, refer to Text-to-speech [EPUB3Overview] .

› 3.3.4 CSS Fonts Level 3

The EPUB 3 CSS Profile includes `@font-face` rules and descriptors as defined in the CSS Fonts Module Level 3 [CSS3Fonts] specification, using syntax as defined in [CSS3Fonts-20110324] and semantics as defined in [CSS3Fonts].

Reading Systems with a CSS Viewport MUST support OpenType [OpenType] and WOFF [WOFF] fonts embedded using the `@font-face` rule.

NOTE

Refer to Embedded Font Intrinsic Fallback [Publications301] for font fallback processing requirements.

In addition, Reading Systems MUST support at least the following `@font-face` font descriptors.

- `font-family`
- `font-style`
- `font-weight`
- `src`
- `unicode-range`

For forwards compatibility with EPUB 2 Reading Systems that do not support `@font-face` rules, authors SHOULD reference a generic font using the `font-family` property.

NOTE

Refer to Resource Obfuscation [OCF301] for Reading System font obfuscation requirements.

› 3.3.5 CSS Text Level 3

The EPUB 3 CSS Profile includes `-epub-` prefixed versions of the following properties from the CSS Text Level 3 [CSS3Text] specification using syntax as defined in [CSS3Text-20110412] and semantics

as defined in [CSS3Text].

- `-epub-hyphens`*
- `-epub-line-break`
- `-epub-text-align-last`
- `-epub-word-break`

* The `-epub-hyphens` property does not include support for the value `all`.

In addition, the EPUB 3 CSS Profile includes the unprefixed `text-transform` property from CSS Text Level 3 using semantics as defined in [CSS3Text] and syntax as defined in [CSS3Text-20110412], with the exception that the `fullwidth` and `fullsize-kana` values are prefixed in the EPUB 3 CSS Profile (`-epub-fullwidth` and `-epub-fullsize-kana`, respectively).

Note that the CSS Text Level 3 module has dropped support for the `fullsize-kana` value since the EPUB 3.0 revision. The EPUB CSS 3 Profile retains this value, but now defines its semantics as below:

`-epub-fullsize-kana`

> This value represents a mapping of HIRAGANA or KATAKANA characters as shown in Appendix B, `-epub-fullsize-kana` *Character Mapping Reference* . This value is typically used for Japanese ruby annotation text.

› 3.3.6 CSS Text Decoration Level 3

The EPUB 3 CSS Profile includes `-epub-` prefixed versions of the following properties from the CSS Text Decoration Level 3 [CSS3TextDecoration] specification using syntax as defined in [CSS3TextDecoration-20130103] and semantics as defined in [CSS3TextDecoration].

- `-epub-text-emphasis`
- `-epub-text-emphasis-color`
- `-epub-text-emphasis-position`
- `-epub-text-emphasis-style`
- `-epub-text-underline-position`

› 3.3.7 CSS Writing Modes

With exceptions for the `direction` and `unicode-bidi` properties as noted below, the EPUB 3 CSS Profile includes all of the features defined in the CSS Writing Modes Module Level 3 [CSS3WritingModes] specification using `-epub-` prefixed property names, syntax as defined in [CSS3WritingModes-20110428] and semantics as defined in [CSS3WritingModes]. Furthermore, as specified in [CSS3WritingModes-20121115], both "`sideways`" and "`mixed`" are permitted as values of the `-epub-text-orientation` property. The values "`vertical-right`", "`rotate-right`", "`rotate-left`", "`rotate-normal`" and "`auto`" of this property are deprecated.

NOTE

The semantics of "vertical-right", "rotate-right", "rotate-left", "rotate-normal", and "auto" is the same as that of "mixed", "sideways-right", "sideways-left", "sideways" and "use-glyph-orientation" in [CSS3WritingModes], respectively.

The -epub-text-combine property is deprecated, and the -epub-text-combine-horizontal property from [CSS3WritingModes-20121115] added.

NOTE

The -epub-text-combine property's values can be mapped to -epub-text-combine-horizontal as follows: 'none' to 'none' and 'horizontal' to 'all'. The syntax 'horizontal <number>' is treated as an error.

The direction and unicode-bidi properties from [CSS3WritingModes] are not included in the EPUB 3 CSS Profile. Authors SHOULD use appropriate [HTML5] markup to express directionality information instead.

› 3.3.8 Selectors

The EPUB 3 CSS Profile includes support for the Selectors Level 3 [Selectors] specification.

› 3.3.9 Media Queries

The EPUB 3 CSS Profile includes @media and @import rules with media queries as defined in the Media Queries [MediaQueries] specification.

› 3.3.10 CSS Namespaces

The EPUB 3 CSS Profile includes the @namespace rule defined in [CSS Namespaces] for declaring the default namespace for a style sheet and for binding prefixes to namespaces.

› 3.3.11 CSS Multi-Column Layout

The EPUB 3 CSS Profile includes all of the features defined in the CSS Multi-column Layout Module [CSSMultiCol] specification with the exception of the column-span property.

CAUTION

Authors should not rely on column behavior in overflow conditions as this behavior is unstable and may change.

CAUTION

Pagination algorithms are not fully defined in CSS. Authors should therefore expect exact pagination points to vary from Reading System to Reading System.

Reading Systems MUST treat the `oeb-column-number` property as an alias for the `column-count` property. The use of the `oeb-column-number` property in EPUB Style Sheets is deprecated; this conformance requirement may be removed in the next major version of EPUB.

3.3.12 Ruby Positioning

The EPUB 3 CSS Profile includes the `-epub-ruby-position` property as defined below:

Name:	-epub-ruby-position
Value:	`over` | `under` | `inter-character`
Initial:	over
Applies to:	ruby text elements
Inherited:	yes
Percentages:	N/A
Media:	visual
Computed value:	as specified

This property controls the placement of ruby text with respect to its base text. Values have the following meanings:

`over`

Ruby text is positioned on the over side of the ruby base.

`under`

Ruby text is positioned on the under side of the ruby base.

`inter-character`

Ruby text is positioned on the right side of the base text. (This value is typically used for Zhuyin Fuhao (Bopomofo) ruby.)

NOTE

The `-epub-ruby-position` property will become an alias for the `ruby-position` property in the CSS Ruby Module [CSS3Ruby].

3.3.13 Display Property Values `oeb-page-head` and `oeb-page-foot` [DEPRECATED]

The `oeb-page-head` and `oeb-page-foot` values are deprecated and expected to be removed or replaced in a future version of EPUB.

Authors MAY continue to include these values in EPUB Style Sheets to define running headers and footers. Refer to the oeb-page-head oeb-page-foot definitions in [ContentDocs30] for more information.

4 PLS Documents

4.1 Overview

This section is informative

The W3C Pronunciation Lexicon Specification [PLS] defines syntax and semantics for XML-based pronunciation lexicons to be used by Automatic Speech Recognition and Text-to-Speech (TTS) engines.

The following sections define conformance criteria for PLS documents when included in EPUB Publications, and rules for associating PLS Documents with XHTML Content Documents.

NOTE

For more information on EPUB 3 features related to synthetic speech, refer to Text-to-speech [EPUB3Overview] .

4.2 EPUB Publication Conformance

A conformant Rendition of an EPUB Publication MUST meet all of the following criteria for inclusion of PLS Documents:

- PLS Documents MAY be associated with XHTML Content Documents. Each XHTML Content Document MAY contain zero or more PLS Document associations.

- PLS Documents MUST be associated with the XHTML Content Document to which it applies using the [HTML5] `link` element with its `rel` attribute set to `pronunciation` and its `type` attribute set to the PLS media type (`application/pls+xml`).

- The `link` element `hreflang` attribute SHOULD be specified on each PLS `link`, and its value MUST match the language for which the pronunciation lexicon is relevant [PLS] when specified.

- PLS Documents MUST meet the content conformance criteria defined in PLS Documents — Content Conformance.

- PLS Documents MUST be represented and located as defined in EPUB Publication — Content Conformance [Publications301] .

› Examples

The following example shows two PLS Documents (one for Chinese and one for Mongolian) associated with an XHTML Content Document.

```
<html … >
    <head>
        …
        <link rel="pronunciation" type="application/pls+xml"
hreflang="zh" href="../speech/zh.pls"/>
        <link rel="pronunciation" type="application/pls+xml"
hreflang="mn" href="../speech/mn.pls"/>
    </head>
    …
</html>
```

› 4.3 Content Conformance

To be considered a Core Media Type Resource, a PLS Document MUST meet all of the following criteria:

Document Properties

› It MUST meet the conformance constraints for XML documents defined in XML Conformance [Publications301] .

› It MUST be valid to the RELAX NG schema for PLS documents available at the URI `http://www.w3.org/TR/pronunciation-lexicon/pls.rng` [PLS].

File Properties

› The PLS Document filename SHOULD use the file extension `.pls`.

› 4.4 Reading System Conformance

A conformant EPUB Reading System MUST meet all of the following criteria for processing PLS Documents:

› Reading Systems with Text-to-Speech (TTS) capabilities SHOULD support PLS.

› Reading Systems that support PLS MUST process PLS documents as defined in [PLS].

› Reading Systems that support PLS MUST apply the supplied pronunciation instructions to all text nodes in the current XHTML Content Document whose language [HTML5] matches the language for which the pronunciation lexicon is relevant [PLS]. The algorithm for matching language tags is defined in BCP47.

› When a pronunciation rule is specified more than once for a given string target in a given language, the last occurrence of the rule takes precedence, in such a way that any previously-defined pronunciation rule gets overridden.

› Reading Systems that support PLS and the SSML Attributes MUST let any pronunciation instructions provided via the `ssml:ph` attribute take precedence in cases where a `pls:grapheme` matches a text node of an element that carries the `ssml:ph` attribute.

› Appendix A. JavaScript epubReadingSystem Object

› A.1 Syntax

```
ReadingSystem = navigator.epubReadingSystem;
```

› A.2 Description

The `epubReadingSystem` object provides an interface through which a Scripted Content Document can query information about a User's Reading System.

The object exposes a number of properties, about the Reading System, such as its name and version, and provides the hasFeature method which can be invoked to determine the features it supports.

Example JavaScript function that displays the name of the current Reading System.

```
alert("Reading System name: " + navigator.epubReadingSystem.name);
```

› A.3 Properties

The following properties MUST be made available for retrieving information about the Reading System.

Required epubReadingSystem properties

Name	Description
`name`	Returns a `String` value representing the name of the Reading System (e.g., "`iBooks`", "`Kindle`").
`version`	Returns a `String` value representing the version of the Reading System (e.g., "`1.0`", "`2.1.1`").
`layoutStyle`	Returns a `String` value representing the style of layout for the content. A Reading System will typically return one of the values "`paginated`" or "`scrolling`", but MAY define values for any additional layout formats it supports.

› A.4 Methods

A.4.1 hasFeature

A.4.1.1 Syntax

```
hasFeature(feature[, version])
```

A.4.1.2 Description

For recognized features, the `hasFeature` method returns a boolean value indicating whether any version is supported.

If the optional `version` parameter is included, the return value indicates support only for the specified version.

The method returns `undefined` if the feature is not recognized by the Reading System.

Example JavaScript function that displays whether the current Reading System supports scripted manipulation of the DOM.

```
var feature = "dom-manipulation";

var conformTest = navigator.epubReadingSystem.hasFeature(feature);

alert("Feature " + feature + " supported?: " + conformTest);
```

A.4.1.3 Features

The following table details the features that MUST be recognized by all Reading Systems that support scripting (spine-level or container-constrained). Reading Systems MAY support some or all of these features (refer to Scripted Content Documents — Reading System Conformance for more information).

Feature names are case-insensitive.

Required epubReadingSystem features

Name	Description
`dom-manipulation`	Scripts MAY make structural changes to the document's DOM (applies to spine-level scripting only).
`layout-changes`	Scripts MAY modify attributes and CSS styles that affect content layout (applies to spine-level scripting only).
`touch-events`	The device supports touch events and the Reading System passes touch events to the content.
`mouse-events`	The device supports mouse events and the Reading System passes mouse events to the content.

Name	Description
`keyboard-events`	The device supports keyboard events and the Reading System passes keyboard events to the content.
`spine-scripting`	Spine-level scripting is supported.

If a Reading System supports a feature defined in this section, it MUST return a `true` value both when queried without the version parameter set and when that parameter is set to the value "`1.0`". Otherwise, it MUST return `false`. Reading System developers SHOULD NOT change the version number of these features independently of this specification.

Additional features MAY be added by Reading System developers, but future versions of this specification MAY append to this list in ways that MAY conflict or be incompatible with any such custom additions.

› Appendix B. `-epub-fullsize-kana` Character Mapping Reference

This appendix is informative

The following table provides character mappings for the `-epub-fullsize-kana` value of the `text-transform` property.

From	From Character	From Name	To	To Character	To Name
03041	ぁ	HIRAGANA LETTER SMALL A	03042	あ	HIRAGANA LETTER A
03043	ぃ	HIRAGANA LETTER SMALL I	03044	い	HIRAGANA LETTER I
03045	ぅ	HIRAGANA LETTER SMALL U	03046	う	HIRAGANA LETTER U
03047	ぇ	HIRAGANA LETTER SMALL E	03048	え	HIRAGANA LETTER E
03049	ぉ	HIRAGANA LETTER SMALL O	0304A	お	HIRAGANA LETTER O
03063	っ	HIRAGANA LETTER SMALL TU	03064	つ	HIRAGANA LETTER TU
03083	ゃ	HIRAGANA LETTER SMALL YA	03084	や	HIRAGANA LETTER YA
03085	ゅ	HIRAGANA LETTER SMALL YU	03086	ゆ	HIRAGANA LETTER YU

03087	ょ	HIRAGANA LETTER SMALL YO	03088	よ	HIRAGANA LETTER YO
0308E	ゎ	HIRAGANA LETTER SMALL WA	0308F	わ	HIRAGANA LETTER WA
03095	ゕ	HIRAGANA LETTER SMALL KA	0304B	か	HIRAGANA LETTER KA
03096	ゖ	HIRAGANA LETTER SMALL KE	03051	け	HIRAGANA LETTER KE
030A1	ァ	KATAKANA LETTER SMALL A	030A2	ア	KATAKANA LETTER A
030A3	ィ	KATAKANA LETTER SMALL I	030A4	イ	KATAKANA LETTER I
030A5	ゥ	KATAKANA LETTER SMALL U	030A6	ウ	KATAKANA LETTER U
030A7	ェ	KATAKANA LETTER SMALL E	030A8	エ	KATAKANA LETTER E
030A9	ォ	KATAKANA LETTER SMALL O	030AA	オ	KATAKANA LETTER O
030C3	ッ	KATAKANA LETTER SMALL TU	030C4	ツ	KATAKANA LETTER TU
030E3	ャ	KATAKANA LETTER SMALL YA	030E4	ヤ	KATAKANA LETTER YA
030E5	ュ	KATAKANA LETTER SMALL YU	030E6	ユ	KATAKANA LETTER YU
030E7	ョ	KATAKANA LETTER SMALL YO	030E8	ヨ	KATAKANA LETTER YO
030EE	ヮ	KATAKANA LETTER SMALL WA	030EF	ワ	KATAKANA LETTER WA
030F5	ヵ	KATAKANA LETTER SMALL KA	030AB	カ	KATAKANA LETTER KA
030F6	ヶ	KATAKANA LETTER SMALL KE	030B1	ケ	KATAKANA LETTER KE
031F0	ク	KATAKANA LETTER SMALL KU	030AF	ク	KATAKANA LETTER KU
031F1	シ	KATAKANA LETTER SMALL SI	030B7	シ	KATAKANA LETTER SI
031F2	ス	KATAKANA LETTER SMALL SU	030B9	ス	KATAKANA LETTER SU
031F3		KATAKANA LETTER SMALL	030C8	ト	KATAKANA LETTER

	ト	TO			TO
031F4	ヌ	KATAKANA LETTER SMALL NU	030CC	ヌ	KATAKANA LETTER NU
031F5	ハ	KATAKANA LETTER SMALL HA	030CF	ハ	KATAKANA LETTER HA
031F6	ヒ	KATAKANA LETTER SMALL HI	030D2	ヒ	KATAKANA LETTER HI
031F7	フ	KATAKANA LETTER SMALL HU	030D5	フ	KATAKANA LETTER HU
031F8	ヘ	KATAKANA LETTER SMALL HE	030D8	ヘ	KATAKANA LETTER HE
031F9	ホ	KATAKANA LETTER SMALL HO	030DB	ホ	KATAKANA LETTER HO
031FA	ム	KATAKANA LETTER SMALL MU	030E0	ム	KATAKANA LETTER MU
031FB	ラ	KATAKANA LETTER SMALL RA	030E9	ラ	KATAKANA LETTER RA
031FC	リ	KATAKANA LETTER SMALL RI	030EA	リ	KATAKANA LETTER RI
031FD	ル	KATAKANA LETTER SMALL RU	030EB	ル	KATAKANA LETTER RU
031FE	レ	KATAKANA LETTER SMALL RE	030EC	レ	KATAKANA LETTER RE
031FF	ロ	KATAKANA LETTER SMALL RO	030ED	ロ	KATAKANA LETTER RO
0FF67	ｧ	HALFWIDTH KATAKANA LETTER SMALL A	0FF71	ｱ	HALFWIDTH KATAKANA LETTER A
0FF68	ｨ	HALFWIDTH KATAKANA LETTER SMALL I	0FF72	ｲ	HALFWIDTH KATAKANA LETTER I
0FF69	ｩ	HALFWIDTH KATAKANA LETTER SMALL U	0FF73	ｳ	HALFWIDTH KATAKANA LETTER U
0FF6A	ｪ	HALFWIDTH KATAKANA LETTER SMALL E	0FF74	ｴ	HALFWIDTH KATAKANA LETTER E
0FF6B	ｫ	HALFWIDTH KATAKANA LETTER SMALL O	0FF75	ｵ	HALFWIDTH KATAKANA LETTER O

0FF6C	ｬ	HALFWIDTH KATAKANA LETTER SMALL YA	0FF94	ｬ	HALFWIDTH KATAKANA LETTER YA
0FF6D	ｭ	HALFWIDTH KATAKANA LETTER SMALL YU	0FF95	ｭ	HALFWIDTH KATAKANA LETTER YU
0FF6E	ｮ	HALFWIDTH KATAKANA LETTER SMALL YO	0FF96	ｮ	HALFWIDTH KATAKANA LETTER YO
0FF6F	ｯ	HALFWIDTH KATAKANA LETTER SMALL TU	0FF82	ｯ	HALFWIDTH KATAKANA LETTER TU

› Appendix C. Acknowledgements and Contributors

This appendix is informative

EPUB has been developed by the International Digital Publishing Forum in a cooperative effort, bringing together publishers, vendors, software developers, and experts in the relevant standards.

The EPUB 3 specifications were prepared by the International Digital Publishing Forum's EPUB Maintenance Working Group, operating under a charter approved by the membership in May, 2010 under the leadership of:

Gylling, Markus (International Digital Publishing Forum (IDPF)) *Chair*
Conboy, Garth (Google Inc.) *Vice-chair*
Duga, Brady (Google Inc.) *Vice-chair, Subgroup Lead*
McCoy, Bill (International Digital Publishing Forum (IDPF)) *Secretary*
Kasdorf, Bill (Apex CoVantage) *Subgroup Lead*
MURATA, Makoto (JEPA EPUB Study Group) *Subgroup Lead*
Sorotokin, Peter (Adobe) *Subgroup Lead*

Active members of the working group included:

› IDPF Members

Abrams, Willie (Ingram Digital)
Acton, Daniel (Google)
Allesi, Ana Maria (HarperCollins)
Amos, Dan (DNAML (DNL eBooks))
Arany, Steve (John Wiley & Sons)
Artin, Michael (Barnes & Noble)
Badger, Brandon (Google)
Ballard, Kevin (Apple Inc.)
Beard, Elliot (HarperCollins)
Belfanti, Paul (Pearson)
Bell, Graham (EDItEUR)
Bell, Jeff (Microsoft)

Bide, Mark (EDItEUR)
Bogaty, Nick (Adobe)
Bowers, Micah (Bluefire Productions)
Brantley, Peter (Internet Archive)
Breglio, Melissa (Apple Inc.)
Broome, Karen (Sony)
Brugge, John (Benetech)
Carbonell, Oliver (Sony)
Chang, Phobos (Chinese Foundation for Digitization Technology)
Chen, Mei-Li (Institute for Information Industry)
Chen, Peter (ITRI)
Choi, Soo (HarperCollins)
Chow, King-Wai (ASTRI (Hong Kong Applied Science & Technology Research Institute))
Clutter, Mat (Random House)
Conboy, Garth (Google)
Cramer, Dave (Hachette Book Group)
Cronin, Margot (Bowker)
Daly, Liza (Threepress)
De Meulemeester, Eric (Jouve/Publishing Dimensions)
DeMeglio, Marisa (DAISY Consortium)
Deltour, Romain (DAISY Consortium)
Dougherty, Casey (Apple Inc.)
Drake, Jama (Impelsys)
Duga, Brady (Google)
Elliott, Ray (Crossway)
Fahlgren, Keith (Threepress)
Fain, Guy (Crossway)
Freese, Eric (Aptara)
Gardeur, Hadrien (Feedbooks)
Gold, Eric (Digital Divide Data)
Goodwin, Jonathan (Appfoundry)
Gopinath, Anith (Impelsys)
Gosling, Andreas (Penguin)
Grazioli, Frank (John Wiley & Sons)
Gunn, Dave (RNIB)
Gylling, Markus (International Digital Publishing Forum (IDPF))
Haas, Matt (Pearson)
Hadfield, Tom (CourseSmart)
Hagino, Masaaki (Voyager Japan)
Hawkins, Kevin (University of Michigan Library)
Hayashi, Junichi (Voyager Japan)
Heiberger, Richard (HarperCollins)
Hepp, Mike (Dartmouth Journal Services)
Herren, Matthew (BlankPage)
Hisashi, Saiga (Sharp)
Hoda, Hisashi (Voyager Japan)
Howard, William (Easypress)
Hughes, Dan (Liguori Publications)
Hulse, Leslie (HarperCollins)
Imsieke, Gerrit (le-tex)
Jain, Anupam (Innodata Isogen)
Jie, Fan (Gansu DUZHE Digital Sci&Tech)
Johnson, Rick (Ingram Digital)
Jung, Kanghee (Incube Technologies)
Kakar, Samir (Aptara)
Kanai, Takeshi (Sony)
Kasdorf, Bill (Apex CoVantage)
Kasher, Bob (BookMasters and Newgen Imaging)

Kato, Kazuyuki (East Co.)
Keating, Patrick (Bluefire Productions)
Kerscher, George (DAISY Consortium)
Kida, Yasuo (Apple Inc.)
Kim, Jean (Barnes & Noble)
Kim, HyunYoung (Incube Technologies)
Kim, Terry (INKA Entworks)
Kitagawa, Masahiro (Impress Holdings)
Koike, Toshiaki (Voyager Japan)
Kok, Dan (Crossway)
Kotrch, Steve (Simon & Schuster)
Larroque, Benoit (Feedbooks)
Levantovsky, Vladimir (Monotype Imaging)
Lu, Cho-Chin (Institute for Information Industry)
Lynch, Ryan (Apple Inc.)
MacFarlane, James (Easypress)
Makower, Dave (Apple Inc.)
Mandelbaum, David (Barnes & Noble)
Manis, Will (Metaplates)
McCloy-Kelley, Liisa (Random House)
McCoy, Bill (IDPF)
Menzies, Tracey (HarperCollins)
Mitchell, Chris (Random House)
Moore, Helen (HarperCollins)
Muller, Eric (Adobe)
Murata, Makoto (JEPA EPUB Study Group)
Mussinelli, Christina (Associazione Italiana Editori)
Nagai, Yoshinori (Sharp)
Novelli, Joe (Sony)
O'Connor, Theresa (Apple Inc.)
Ohmura, Yoshinori (Impress Holdings)
Olenick, Michael (Bowker)
Oshiyama, Taka (East Co.)
Pagano, Pat (Barnes & Noble)
Picco, Marty (AppFoundry)
Prabhu, John (HOV Services)
Pritchett, James (Learning Ally (formerly RFB&D))
Rao, Vishnu (Sharp Laboratories)
Rivlin, John (Google)
Rubino, Frank (Kaplan Publishing)
Ruffino, Daniel (Penguin)
Rui Hua, Wang (Gansu DUZHE Digital Sci&Tech)
Ruse, Tyler (LibreDigital)
Sanicola, Daniel (Penguin)
Schirmer, Lorenz (Monotype Imaging)
Severdia, Ron (Metrodigi, Inc.)
Shiohama, Daihei (Voyager Japan)
Shrivastava, Abhishek (CourseSmart)
Siegman, Tzviya (John Wiley & Sons)
Slavin, Wayne (Barnes & Noble)
Slye, Christopher (Adobe)
Smith, Michael (IDPF)
Soiffer, Neil (Design Science)
Sorotokin, Peter (Adobe)
Stevenson, Tobias (eBookArchitects)
Tahara, Kyoji (Toppan Printing)
Takase, Hiroshi (East Co.)
Tallent, Joshua (eBookArchitects)

Tanabe, Shu (Toppan Printing)
Thomas, Vinu (Impelsys)
Tsumagari, Koichiro (Voyager Japan)
Valentine, Chelsea (LibreDigital)
Vangage, Peter (Harlequin)
Vido, Ariel (Geografica Editora)
Wait, John (Pearson)
Walkley, George (Hachette Book Group)
Watters, Kevin (Harlequin)
Webster, Roger (Barnes & Noble)
Weck, Daniel (DAISY Consortium)
Wei, Selena (Chinese Foundation for Digitization Technology)
White, Russell (Random House)
Wiles, Alexis (Overdrive, Inc.)
Witwer, Adam (O'Reilly)
Wright, Ric (Adobe)
Young, Liz (Crossway)
Zu, Alex (ASTRI (Hong Kong Applied Science & Technology Research Institute))

› Invited Experts/Observers

Bowes, Rick
Cazenove, Rhys
Collingridge, Peter
Cook, Mike
Etemad, Elika J. *W3C CSS WG Liason*
Forster, Karen
Freed, Geoff
Fujisawa, Jun
Garrish, Matt
Gould, Bryan
Görner, Martin
Hsieh, Michael
Ishii, Koji
Johar, Kenny
Karlsson, Mattias
Kennedy, Dianne
Kilborn, Peter
Koppel, Josh
Lee, Tommy
Lu, Kenny
Lubeck, Scott
Masanori, Kusunoki
McKinney, Steven
Murakami, Shinyu
Ning, Elliott
Noring, Jon
Norton, Paul
Oishi, Yasuharu
Passey, Lee
Rosmaita, Gregory
Seaman, David
Shan, Walter
Smith(tm), Michael (W3C) *W3C Liason*
Sperberg, Roger
Walsh, Norman

Zergaoui, Mohamed

For more detailed acknowledgements and information about contributors to each version of EPUB, refer to Acknowledgements and Contributors [EPUB3Overview] .

› References

Normative References

[AltStyleTags] *Alternate Style Tags* . Elika J. Etemad.

[CSS Namespaces] *CSS Namespaces Module* . Elika J. Etemad, et al.

[CSS2.0] *Cascading Style Sheets, level 2 - CSS2 Specification* . Bert Bos, et al. 12 May 1998 (revised 11 April 2008).

[CSS2.1] *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification* . Bert Bos, et al. 7 June 2011.

[CSS3Fonts] *CSS Fonts Module Level 3* . John Daggett.

[CSS3Fonts-20110324] *CSS Fonts Module Level 3 (20110324)* . John Daggett. 24 March 2011.

[CSS3Ruby] *CSS3 Ruby Annotation Module* .

[CSS3Speech] *CSS3 Speech Module* . Dave Raggett, et al.

[CSS3Speech-20110818] *CSS3 Speech Module (20110818)* . Dave Raggett, et al. 19 April 2011.

[CSS3Text] *CSS Text Level 3* . Elika J. Etemad, et al.

[CSS3Text-20110412] *CSS Text Level 3 (20110412)* . Elika J. Etemad, et al. 12 April 2011.

[CSS3TextDecoration] *CSS Text Decoration Module Level 3* . Elika J. Etemad, et al.

[CSS3TextDecoration-20130103] *CSS Text Decoration Module Level 3 (20130103)* . Elika J. Etemad, et al. 3 January 2013.

[CSS3WritingModes] *CSS Writing Modes Module Level 3* . Elika J. Etemad, et al.

[CSS3WritingModes-20110428] *CSS Writing Modes Module Level 3 (20110428)* . Elika J. Etemad, et al. 28 April 2011.

[CSS3WritingModes-20121115] *CSS Writing Modes Module Level 3 (20121115)* . Elika J. Etemad, et al. 15 November 2012.

[CSSMultiCol] *CSS Multi-column Layout Module* . Håkon Wium Lie.

[ContentDocs30] *EPUB Content Documents 3.0* .

[ContentDocs301] *EPUB Content Documents 3.0.1* .

[HTML+RDFa11] *HTML+RDFa 1.1* . Support for RDFa in HTML4 and HTML5. Manu Sporny. 22 August 2013.

[HTML5] *HTML5: A vocabulary and associated APIs for HTML and XHTML* .

[MATHML] *Mathematical Markup Language (MathML) Version 3.0* . David Carlisle, et al. 21 October 2010.

[MediaOverlays301] *EPUB Media Overlays 3.0.1* .

[MediaQueries] *Media Queries* .

[Microdata] *HTML Microdata (20121025)* . Ian Hickson. 25 October 2012.

[OCF301] *Open Container Format 3.0.1* .

[OPF2] *Open Packaging Format 2.0.1* .

[OpenType] *ISO/IEC 14496-22:2009 - Information technology -- Coding of audio-visual objects -- Part 22: Open Font Format* .

[PLS] *Pronunciation Lexicon Specification 1.0 (PLS)* . Paolo Baggia. 14 October 2008.

[Publications301] *EPUB Publications 3.0.1* .

[RDFa11] *RDFa Core 1.1 - Second Edition* . Syntax and processing rules for embedding RDF through attributes. Ben Adida, et al. 22 August 2013.

[RFC2046] *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types* *(RFC 2046)* . N. Freed, N. Borenstein. November 1996.

[RFC2119] *Key words for use in RFCs to Indicate Requirement Levels* *(RFC 2119)* . March 1997.

[RFC5646] *Tags for Identifying Languages* *(RFC 5646)* . A. Phillips, M. Davis. September 2009.

[SSML] *Speech Synthesis Markup Language (SSML) Version 1.1* . Daniel C. Burnett, et al. 7 September 2010.

[SVG] *Scalable Vector Graphics (SVG) 1.1 (Second Edition)* . Erik Dahlström, et al. 09 June 2011.

[SVG Access] *Accessibility Features of SVG* . Charles McCathieNevile, et al. 7 August 2000.

[Selectors] *Selectors Level 3* .

[StructureVocab] *EPUB 3 Structural Semantics Vocabulary* .

[Unicode] *The Unicode Consortium.* *The Unicode Standard.* .

[WAI-ARIA] *Accessible Rich Internet Applications (WAI-ARIA) 1.0* . James Craig, et al.

[WAI-ARIA-1.1] *Accessible Rich Internet Applications (WAI-ARIA) 1.1* . James Craig, et al.

[WCAG20] *Web Content Accessibility Guidelines (WCAG) 2.0* . Ben Caldwel, et al. 11 December 2008.

[WOFF] *WOFF File Format 1.0* . Jonathan Kew, et al.

[XML] *Extensible Markup Language (XML) 1.0 (Fifth Edition)* . T. Bray, et al. 26 November 2008.

[XML Events] *XML Events* . Shane McCarron, et al. 14 October 2003.

[XMLNS] *Namespaces in XML (Third Edition)* . T. Bray, D. Hollander, A. Layman, R. Tobin. W3C. 8 December 2009.

Informative References

[refIPA] *http://www.langsci.ucl.ac.uk/ipa/ipachart.html* . . 2005.

[EPUB3Changes] *EPUB 3.0.1 Differences from EPUB 3.0* . Markus Gylling, et al.

[EPUB3Overview] *EPUB 3 Overview* . Garth Conboy, et al.

[MetaTags] *Supported Meta Tags* .

[Role] *Role Attribute* . An attribute to support the role classification of elements. McCarron, et al. 05 August 2010.

Bureau of Indian Standards

BIS is a statutory institution established under the *Bureau of Indian Standards Act*, 2016 to promote harmonious development of the activities of standardization, marking and quality certification of goods and attending to connected matters in the country.

Copyright

Review of Indian Standards

Amendments are issued to standards as the need arises on the basis of comments. Standards are also reviewed periodically; a standard along with amendments is reaffirmed when such review indicates that no changes are needed; if the review indicates that changes are needed, it is taken up for revision. Users of Indian Standards should ascertain that they are in possession of the latest amendments or edition by referring to the website-www.bis.gov.in or www.standardsbis.

This Indian Standard has been developed from Doc No.: LITD 15 (18265).

Amendments Issued Since Publication

Amend No.	Date of Issue	Text Affected

BUREAU OF INDIAN STANDARDS

Headquarters:

Manak Bhavan, 9 Bahadur Shah Zafar Marg, New Delhi 110002
Telephones: 2323 0131, 2323 3375, 2323 9402 *Website*: www.bis.gov.in

Regional Offices:		*Telephones*
Central	: 601/A, Konnectus Tower -1, 6th Floor, DMRC Building, Bhavbhuti Marg, New Delhi 110002	2323 7617
Eastern	: 8th Floor, Plot No 7/7 & 7/8, CP Block, Sector V, Salt Lake, Kolkata, West Bengal 700091	2367 0012 2320 9474
Northern	: Plot No. 4-A, Sector 27-B, Madhya Marg, Chandigarh 160019	265 9930
Southern	: C.I.T. Campus, IV Cross Road, Taramani, Chennai 600113	2254 1442 2254 1216
Western	: Plot No. E-9, Road No.-8, MIDC, Andheri (East), Mumbai 400093	2821 8093

Branches : AHMEDABAD. BENGALURU. BHOPAL. BHUBANESHWAR. CHANDIGARH. CHENNAI. COIMBATORE. DEHRADUN. DELHI. FARIDABAD. GHAZIABAD. GUWAHATI. HIMACHAL PRADESH. HUBLI. HYDERABAD. JAIPUR. JAMMU & KASHMIR. JAMSHEDPUR. KOCHI. KOLKATA. LUCKNOW. MADURAI. MUMBAI. NAGPUR. NOIDA. PANIPAT. PATNA. PUNE. RAIPUR. RAJKOT. SURAT. VISAKHAPATNAM.

Published by BIS, New Delhi