INTERNATIONAL STANDARD

ISO/IEC 23736-6

First edition
2020-02

Information technology — Digital publishing — EPUB 3.0.1 —

Part 6: **Canonical fragment identifiers**

Technologies de l'information — Publications numériques — EPUB 3.0.1 —

Partie 6: Identificateurs de fragment canoniques

Reference number
ISO/IEC 23736-6:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by the World Wide Web Consortium (W3C) (as EPUB Canonical Fragment Identifiers 1.1) and drafted in accordance with its editorial rules. It was adopted, under the JTC 1 PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

A list of all parts in the ISO/IEC 23736 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

EPUB Canonical Fragment Identifiers 1.1

Recommended Specification 5 January 2017

This version

http://www.idpf.org/epub/linking/cfi/epub-cfi-20170105.html

Latest version

http://www.idpf.org/epub/linking/cfi/epub-cfi.html

Previous version

http://www.idpf.org/epub/linking/cfi/epub-cfi-20161130.html

Previous recommendation

http://www.idpf.org/epub/linking/cfi/epub-cfi-20140628.html

Document history

Changes to this document
Issues addressed in this revision
Report an issue
Errata

Editors

Peter Sorotokin, Adobe

Garth Conboy, Google Inc.

Brady Duga, Google Inc.

John Rivlin, Google Inc.

Don Beaver, Apple Inc.

Kevin Ballard, Apple Inc.

Alastair Fettes, Apple Inc.

Daniel Weck, DAISY Consortium

Status of this Document

This section describes the status of this document at the time of its publication. Other documents might supersede this document.

This document was produced by the EPUB Working Group under the EPUB Working Group Charter approved on 8 July 2015.

This document has been reviewed by the IDPF membership and is endorsed by the IDPF Board as a Recommended Specification. This document is considered stable and can be referenced from other specifications and documents.

Feedback on this document can be provided to the EPUB Working Group's mailing list or issue tracker.

This document is governed by the IDPF Policies and Procedures.

Table of Contents

› 1 Overview

› 1.1 Purpose and Scope

This section is informative

This specification, EPUB Canonical Fragment Identifier (epubcfi), defines a standardized method for referencing arbitrary content within an EPUB® Publication through the use of fragment identifiers.

The Web has proven that the concept of hyperlinking is tremendously powerful, but EPUB Publications have been denied much of the benefit that hyperlinking makes possible because of the lack of a standardized scheme to link into them. Although proprietary schemes have been developed and implemented for individual Reading Systems, without a commonly-understood syntax there has been no way to achieve cross-platform interoperability. The functionality that can see significant benefit from breaking down this barrier, however, is varied: from reading location maintenance to annotation attachment to navigation, the ability to point into any Publication opens a whole new dimension not previously available to developers and Authors.

This specification attempts to rectify this situation by defining an arbitrary structural reference that can uniquely identify any location, or simple range of locations, in an EPUB Publication: the EPUB CFI. The following considerations have strongly influenced the design and scope of this scheme:

- The mechanism used to reference content should be interoperable: references to a reading position created by one Reading System should be usable by another.
- Document references to EPUB content should be enabled in the same way that existing hyperlinks enable references throughout the Web.
- Each location in an EPUB file should be able to be identified without the need to modify the document.
- All fragment identifiers that reference the same logical location should be equal when compared.
- Comparison operations, including tests for sorting and comparison, should be able to be performed without accessing the referenced files.
- Simple manipulations should be possible without access to the original files (e.g., given a reference deep in a file, it should be possible to generate a reference to the start of the file).
- Identifier resolution should be reasonably efficient (e.g., processing of the first chapter is not necessary to resolve a fragment identifier that points to the last chapter).
- References should be able to recover their target locations through parser variations and document revisions.
- Expression of simple, contiguous ranges should be supported.
- An extensible mechanism to accommodate future reference recovery heuristics should be provided.

In the case of both Standard EPUB CFIs and Intra-Publication EPUB CFI, this specification conforms with the guidelines expressed by W3C in Section 6. Best Practices for Fragid Structures [FragIDBestPractices].

In other words, both standard CFI URIs (e.g., "`book.epub#epubcfi(…)`", referred media type "`application/epub+zip`") and intra-publication CFI URIs (e.g., "`package.opf#epubcfi(…)`", referred media type "`application/oebps-package+xml`") make use of a fragment identifier syntax that does not overlap with existing schemes in the context of the aforementioned media types' suffix registrations (i.e., "`-xml`" and "`-zip`").

› 1.2 Terminology

Please refer to [EPUB 3.1] for definitions of EPUB-specific terminology used in this document.

Standard EPUB CFI

A publication-level EPUB CFI links into an EPUB Publication. The path preceding the EPUB CFI references the location of the EPUB Publication.

Intra-Publication EPUB CFI

An intra-publication EPUB CFI allows one Content Document to reference another within the same Rendition of an EPUB Publication. The path preceding the EPUB CFI references the current Rendition's Package Document.

Refer to Intra-Publication CFIs for more information.

› 1.3 Typographic Conventions

The following typographic conventions are used in this specification:

`markup`

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

`markup link`

Links to markup and code definitions are in underlined red monospace font.

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

URIs are in navy blue monospace font.

hyperlink

Hyperlinks are underlined and blue.

[reference]

Normative and informative references are enclosed in square brackets.

Term

Terms defined in the Terminology are in capital case.

Term Link

Links to term definitions have a dotted blue underline.

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

```
Informative markup examples are in light gray boxes.
```

NOTE

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

CAUTION

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

1.4 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 and appendixes 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 appendixes applies to all child content and subsections they contain.

All examples in this specification are informative.

2 EPUB CFI Definition

2.1 Introduction

This section is informative

A fragment identifier is the part of an IRI [RFC3987] that defines a location within a resource. Syntactically, it is the segment attached to the end of the resource IRI starting with a hash (`#`). For HTML documents, IDs and named anchors are used as fragment identifiers, while for XML documents the Shorthand XPointer [XPTRSH] notation is used to refer to a given ID.

A Canonical Fragment Identifier (CFI) is a similar construct to these, but expresses a location within an EPUB Publication. For example:

```
book.epub#epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/3:10)
```

The function-like string immediately following the hash (`epubcfi(…)`) indicates that this fragment identifier conforms to the scheme defined by this specification, and the value contained in the parentheses is the syntax used to reference the location within the specified EPUB Publication (`book.epub`). Using the processing rules defined in Path Resolution, any Reading System can parse this syntax, open the corresponding Content Document in the EPUB Publication and load the specified location for the user.

A complete definition of the EPUB CFI syntax is provided in the next section.

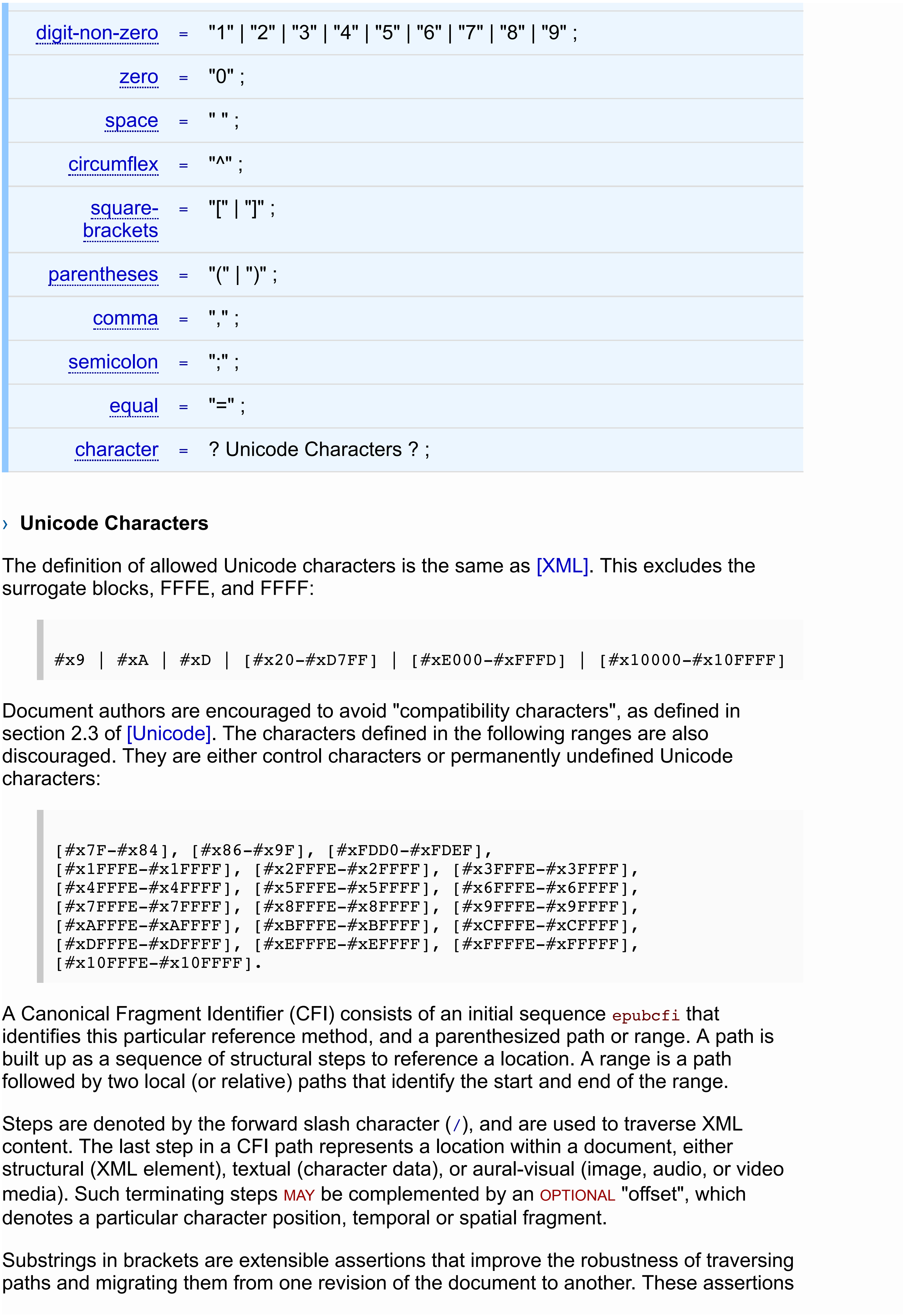
NOTE

epub has been prepended to the name of the scheme, as a more generic CFI-like scheme might be defined in the future for all XML+ZIP-based file formats.

› 2.2 Syntax

(EBNF productions ISO/IEC 14977)
All terminal symbols are in the Unicode Block 'Basic Latin' (U+0000 to U+007F).

```
fragment = "epubcfi(" , ( path , [ range ] ) , ")" ;
path = step , local_path ;
range = "," , local_path , "," , local_path ;
local_path = { step } , ( redirected_path | [ offset ] );
redirected_path = "!" , ( offset | path );
step = "/" , integer , [ "[" , assertion , "]" ] ;
offset = ( ( ":" , integer ) | ( "@" , number , ":" , number ) | ( "~" , number , [ "@" , number , ":" , number ] ) ) , [ "[" , assertion , "]" ] ;
number = ( digit-non-zero , { digit } , [ "." , { digit } , digit-non-zero ] ) | ( zero , [ "." , { digit } , digit-non-zero ] ) ;
integer = zero | ( digit-non-zero , { digit } ) ;
assertion = ( ( value , [ "," , value ] ) | ( "," , value ) | ( parameter ) ) { parameter } ;
parameter = ";" , value-no-space , "=" , csv ;
csv = value , { "," , value } ;
value = string-escaped-special-chars ;
value-no-space = value - ( [ value ] , space , [ value ] ) ;
special-chars = circumflex | square-brackets | parentheses | comma | semicolon | equal ;
escaped-special-chars = ( circumflex , circumflex ) | ( circumflex , square-brackets ) | ( circumflex , parentheses ) | ( circumflex , comma ) | ( circumflex , semicolon ) | ( circumflex , equal ) ;
character-escaped-special = ( character - special-chars ) | escaped-special-chars ;
string-escaped-special-chars = character-escaped-special , { character-escaped-special } ;
digit = zero | digit-non-zero ;
```



digit-non-zero	=	"1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
zero	=	"0" ;
space	=	" " ;
circumflex	=	"^" ;
square-brackets	=	"[" | "]" ;
parentheses	=	"(" | ")" ;
comma	=	"," ;
semicolon	=	";" ;
equal	=	"=" ;
character	=	? Unicode Characters ? ;

› Unicode Characters

The definition of allowed Unicode characters is the same as [XML]. This excludes the surrogate blocks, FFFE, and FFFF:

```
#x9 | #xA | #xD | [#x20-#xD7FF] | [#xE000-#xFFFD] | [#x10000-#x10FFFF]
```

Document authors are encouraged to avoid "compatibility characters", as defined in section 2.3 of [Unicode]. The characters defined in the following ranges are also discouraged. They are either control characters or permanently undefined Unicode characters:

```
[#x7F-#x84], [#x86-#x9F], [#xFDD0-#xFDEF],
[#x1FFFE-#x1FFFF], [#x2FFFE-#x2FFFF], [#x3FFFE-#x3FFFF],
[#x4FFFE-#x4FFFF], [#x5FFFE-#x5FFFF], [#x6FFFE-#x6FFFF],
[#x7FFFE-#x7FFFF], [#x8FFFE-#x8FFFF], [#x9FFFE-#x9FFFF],
[#xAFFFE-#xAFFFF], [#xBFFFE-#xBFFFF], [#xCFFFE-#xCFFFF],
[#xDFFFE-#xDFFFF], [#xEFFFE-#xEFFFF], [#xFFFFE-#xFFFFF],
[#x10FFFE-#x10FFFF].
```

A Canonical Fragment Identifier (CFI) consists of an initial sequence `epubcfi` that identifies this particular reference method, and a parenthesized path or range. A path is built up as a sequence of structural steps to reference a location. A range is a path followed by two local (or relative) paths that identify the start and end of the range.

Steps are denoted by the forward slash character (`/`), and are used to traverse XML content. The last step in a CFI path represents a location within a document, either structural (XML element), textual (character data), or aural-visual (image, audio, or video media). Such terminating steps MAY be complemented by an OPTIONAL "offset", which denotes a particular character position, temporal or spatial fragment.

Substrings in brackets are extensible assertions that improve the robustness of traversing paths and migrating them from one revision of the document to another. These assertions preserve additional information about traversed elements of the document, which makes it possible to recover intended location even after some modifications are made to the EPUB Publication.

Although the **value** definition in the syntax above allows any a sequence of characters, a circumflex (`^`) MUST be used to escape the following characters to ensure their presence does not interfere with parsing:

- brackets (`[`,`]`)
- circumflex (`^`)
- comma (`,`)
- parentheses (`(`,`)`)
- semicolon (`;`)

Example of an EPUB CFI that points to a location after the text "`2[1]`*".*

```
epubcfi(/6/14[chap05ref]!/4[body01]/10/2/1:3[2^[1^]])
```

The following rules apply to the use of numbers and integers within the path or range:

- leading zeros are not allowed for numbers or integers (to ensure uniqueness);
- trailing zeros are not allowed in the fractional part of a number;
- zero MUST be represented as the integer `0`;
- numbers in the range `1 > N > 0` MUST have a leading `0.`;
- integral numbers MUST be represented as integers.

2.3 Character Escaping

As described in Syntax, the EPUB CFI grammar contains characters that have a special purpose as delimiters within a fragment identifier expression. These characters MUST be escaped using the circumflex '`^`' character when not intended for use as delimiters, so that they can appear within the EPUB CFI data without being mistaken for delimiters. Depending on the usage context of such EPUB CFI, further character escaping MAY be necessary in order to ensure that all potentially-conflicting text tokens are encoded correctly.

- IRI and URI references:
  - The EPUB CFI (fragment identifier) scheme is designed to be used within URI and IRI references. The [RFC3986] specification defines a number of "reserved" characters that have a specific purpose as delimiters, and which MAY need to be escaped in cases when they would otherwise conflict with the syntactical structure of the URI/IRI reference. The character used for escaping is the percent sign '`%`', and escapable characters get percent-encoded. For example, the percent character itself becomes "`%25`" when it gets escaped

(note the difference with EPUB CFI's circumflex '`^`', which gets escaped using a double character '`^^`').

  - Unlike IRI references, URI references require unicode characters to be ASCII-encoded. Although the EPUB specification itself is based on IRIs (i.e. authors and production tools are expected to use IRIs), some systems or APIs might only support URIs. As a result, implementors MAY still need too handle the conversion of IRI to URI references, as defined in [RFC3987]. Disallowed characters are escaped as follows:

    - Each disallowed character is converted to UTF-8 [RFC2279] as one or more bytes. The disallowed characters in URI references include all non-ASCII characters, plus the excluded characters listed in Section 2.4 of [RFC2396], except for the number sign '`#`' and percent sign '`%`' and the square bracket characters re-allowed in [RFC2732].

      The resulting bytes are escaped with the URI escaping mechanism (that is, converted to '`%HH`', where HH is the hexadecimal notation of the byte value).

      The original character is replaced by the resulting character sequence.

- (X)HTML context:

  IRI references are designed to be used in the various types of documents that EPUB Publications comprise. XML and (X)HTML represent yet another insertion context that requires specific character escaping rules. For example, double quote characters or angle brackets conflict with significant delimiters in the markup syntax, and MUST therefore be escaped using the `&xxx;` special sequence (character reference).

When multiple layers of character escaping are applied to escape or unescape an EPUB CFI, they MUST be applied in reverse order to revert back to the original form. For example, [EPUB-CFI -> IRI -> (X)HTML] becomes [(X)HTML -> IRI -> EPUB-CFI]

The following example shows an EPUB CFI in its "raw" form (only with '`^`' circumflex escaping). Note the assertion text at the end of it, with escaped square brackets as well as the escaped circumflex character itself (the unescaped text is 'Φ-"spa ce"-99%-aa[bb]^'):

```
epubcfi(/6/4!/4/10/2/1:3[Φ-"spa ce"-99%-aa^[bb^]^^])
```

When taking part in an IRI, the space character within the assertion might become percent-escaped ('`%20`'), and the percent character itself MUST be escaped ('`%25`'). Note that the square brackets '`[`' '`]`' and semicolumn '`:`' are "reserved" characters (as per the URI specification) but because they serve no purpose as delimiters when the IRI processor extracts the fragment identifier, they do not need to be escaped (i.e. the fragment component of the IRI can non-ambiguously be parsed by processing all the text after the '`#`' character). The circumflex '`^`' also falls within the category of "unwise" (or "unsafe") characters, but the EPUB fragment identifier scheme does not require escaping them. Here is the IRI-escaped EPUB CFI:

```
#epubcfi(/6/4!/4/10/2/1:3[Φ-"spa%20ce"-99%25-aa^[bb^]^^])
```

When the IRI appears within an XML attribute, the double quote character (quotation mark) is significant as a delimiter of the attribute value, so it becomes escaped with '`&#x22;`'. Note that the Cyrillic "EF" character ('Φ') is directly supported in EPUB XML documents (which use the

UTF-8 encoding to represent the unicode character repertoire), so it doesn't need to be encoded:

```
#epubcfi(/6/4!/4/10/2/1:3[Φ-&#x22;spa%20ce&#x22;-99%25-aa^[bb^]^^])
```

If the IRI need to be converted to URI, the non-ASCII Cyrillic "EF" character ('Φ') would get percent-escaped with 2 bytes ('`0xd0 0xa4`', in hexadecimal). This would result in the following URI:

```
#epubcfi(/6/4!/4/10/2/1:3[%d0%a4-%22spa%20ce%22-99%25-aa^[bb^]^^])
```

URI encoding / decoding APIs usually "aggressively" percent-encode characters, as demonstrated in the following example. Note how the circumflexes '`^`' (%5E), square brackets '`[`' (%5B) '`]`' (%5D) and double-quotes '`"`' (%22) are also percent-encoded (due to their "unsafe" / "unwise" nature within URIs) :

```
#epubcfi(/6/4!/4/10/2/1:3%5B%D0%A4-%22spa%20ce%22-99%25-
aa%5E%5Bbb%5E%5D%5E%5E%5D)
```

› 3 EPUB CFI Processing

› 3.1 Path Resolution

The process of resolving an EPUB CFI to a location within an EPUB Publication begins with the root `package` element of the Package Document. Each step in the CFI is then processed one by one, left to right, applying the rules defined in the following subsections.

NOTE

The EPUB CFI examples in the following subsections are based on the sample documents in Examples.

› 3.1.1 Step Reference to Child Element or Character Data (`/`)

A step with a slash (`/`) followed by a positive integer refers to either a child element or a chunk of character data, as per the rules defined herein:

- [XML] content other than element and character data is ignored. Note that as per the [XML] specification, character data inside CDATA sections is included, and conversely, XML comments are ignored.
- [XML] character data that corresponds to insignificant white space (typically used for markup formatting/indenting) is preserved. Character and entity references are considered expanded, and character data is obtained from the "included replacement text" (as per the terminology defined in the [XML] specification).
- [XML] character data that is interspersed amongst sibling child elements (i.e., "mixed content" context) is logically organised into (potentially-empty) chunks of contiguous character data: the first chunk is located before the first child element (left sibling), the last chunk is located after the last child element (right sibling), and there is one

chunk between each pair of child elements. When there are no child elements, there is one (potentially-empty) chunk of character data. Consecutive (potentially-empty) chunks of character data are each assigned odd indices (i.e., starting at 1, followed by 3, etc.).

- Child [XML] elements are assigned even indices (i.e., starting at 2, followed by 4, etc.). Additionally, 0 is a valid index that refers to a non-existing element which virtually precedes the first potentially-empty chunk of character data within the parent element's content. Similarly, `n+2` is a valid index that refers to a non-existing element which virtually follows the last potentially-empty chunk of character data, where `n` is the even index of the last child element, or 0 if there are no child elements. CFI processors (e.g., Reading Systems) MUST be capable of consuming (e.g., parsing and interpreting) CFI expressions containing references to the 0 and `n+2` "virtual" elements, even when the first (or last, respectively) chunk of character data is empty. Conversely, the *production* of such CFI expressions is governed by the following conformance requirement: if the first chunk of character data is empty, a CFI expression SHOULD NOT be constructed using a reference to the "virtual" element at index 0, instead the "real" first child element (at index 2) SHOULD be referred to. Similarly, if the last chunk of character data is empty, a CFI expression SHOULD NOT be constructed using a reference to the "virtual" element at index `n+2`, instead the "real" last child element (at index `n`) SHOULD be referred to.

> NOTE
>
> The "virtual" first / last elements mechanism might facilitate interoperability with certain instances of DOM Ranges, whereby non-existing elements are used to span across textual content without relying on character offsets at the start/end boundaries.

For a Standard EPUB CFI, the leading step in the CFI MUST start with a slash (`/`) followed by an even number that references the `spine` child element of the Package Document's root `package` element. The Package Document traversed by the CFI MUST be the one specified as the Default Rendition in the EPUB Publication's `META-INF/container.xml` file (i.e., the Package Document referenced by the first `rootfile` element in `container.xml`).

For an Intra-Publication EPUB CFI, the first step MUST start with a slash followed by a node number that references a position in Package Document starting from the root `package` element.

› 3.1.2 XML ID Assertion (`[`)

When an EPUB CFI references an element that contains an ID [XML], the corresponding path step MUST include that ID in square brackets (i.e., after the slash (`/`) and even number that identifies the element).

Specification of identifiers adds robustness to the CFI scheme: a Reading System can determine that the location referenced by the CFI is not the original intended location, and can use the identifier to compute the set of steps that reach the desired destination in the content (see Intended Target Location Correction). The cost of this added robustness is that comparison (and sorting) of CFI strings can be performed only after logically stripping all bracketed substrings (see Sorting Rules).

› 3.1.3 Step Indirection (`!`)

If a step, or a sequence of steps, points to an element that references another document, the exclamation mark (`!`) MUST be used whenever that step is immediately followed by an expression that applies to the referenced document ("indirection"). The following expression is then resolved from the root element of the referenced XML document, or from the targeted XML fragment (when specified).

Only the following references are honored:

- For `itemref` in the Package Document `spine`, the reference is defined by the `href` attribute of the corresponding `item` element in the `manifest` (i.e., that the `itemref`'s `idref` attribute references).
- For [HTML] `iframes` and `embed` elements, references are defined by the `src` attribute
- For the [HTML] `object` element, the reference is defined by the `data` attribute
- For [SVG] `image` and `use` elements, references are defined by the `xlink:href` attribute

> NOTE
>
> This scheme does not take into account hyperlinks, only embedding references. Consequently, it is illegal to follow links from the [HTML] (or [SVG]) `a` element.

› 3.1.4 Character Offset (`:`)

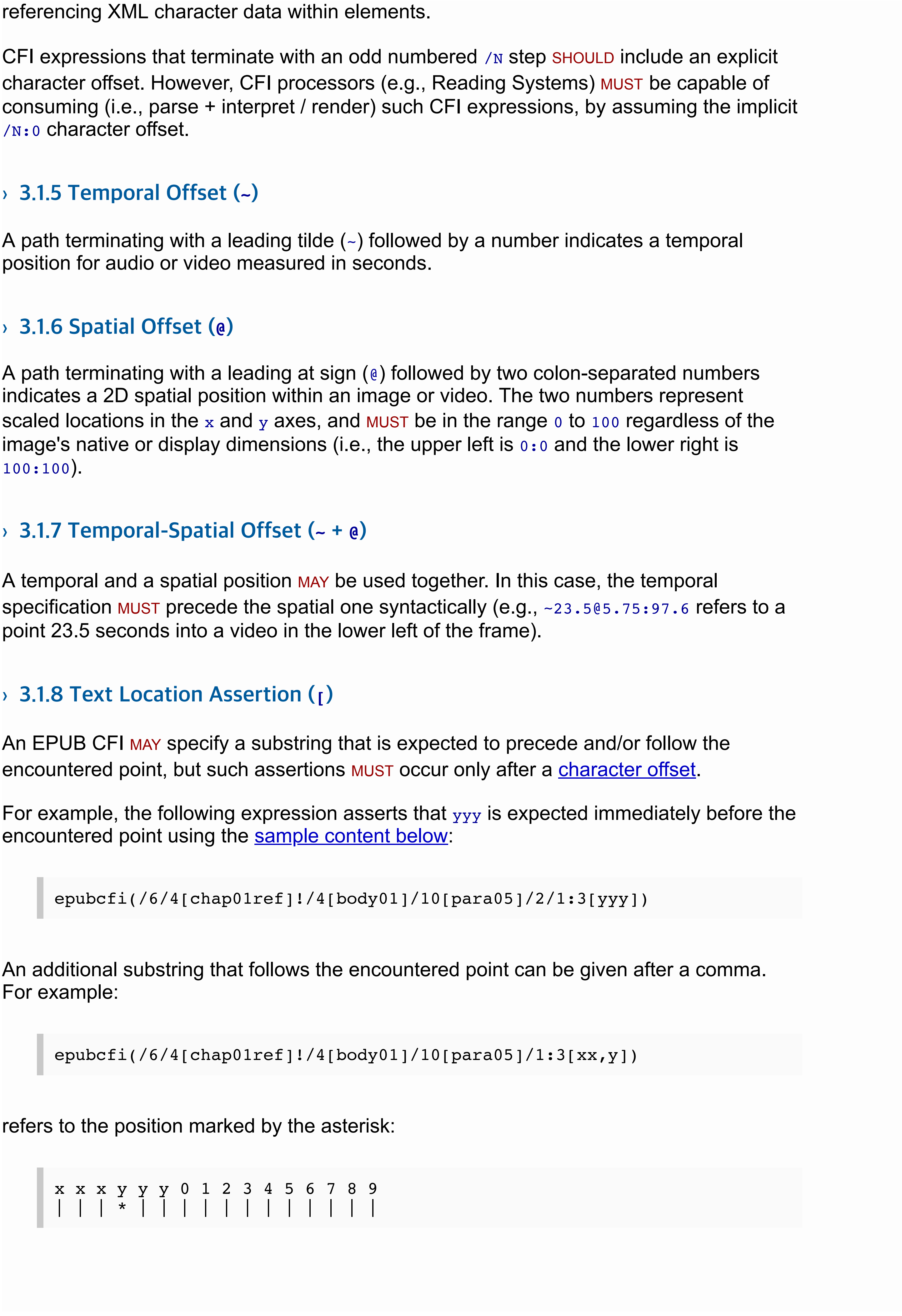
A path terminating with a leading colon (`:`) followed by an integer refers to a character offset. The given character offset MAY apply to an element only if this element is the [HTML] `img` element with an `alt` attribute containing the text to which the character offset applies.

For XML character data, the offset is zero-based and always refers to a position between characters, so `0` means before the first character and a number equal to the total UTF-16 length means after the last character. A character offset value greater than the UTF-16 length of the available text MUST NOT be specified.

In this specification, the definition of an "offset" within XML character data is based on the UTF-16 text encoding, whereby each "character" (Unicode code point) MAY be represented using a single 16-bit code unit, or two units (surrogate pairs, for Unicode characters outside of BMP / Basic Multilingual Plane) [Unicode]. A CFI "character offset" is a zero-based number that refers to a position between UTF-16 code units. Here, the "length" of the text is the total count of 16-bit units. Offset zero therefore means before the first 16-bit unit, and a number equal to the "length" of the text means after the last 16-bit unit. An offset value greater than the "length" of the text MUST NOT be specified.

> NOTE
>
> Counting the number of text "characters" based on UTF-16 code units (instead of Unicode code points) is compatible with the DOM Range model [DOM2 Traversal Range], and with the String API [ECMA-262]

A character offset MAY follow a `/N` step. For XHTML Content Documents, `N` would be an even number when referencing the `alt` text of an `img` element, and `N` would be odd when



referencing XML character data within elements.

CFI expressions that terminate with an odd numbered `/N` step SHOULD include an explicit character offset. However, CFI processors (e.g., Reading Systems) MUST be capable of consuming (i.e., parse + interpret / render) such CFI expressions, by assuming the implicit `/N:0` character offset.

› 3.1.5 Temporal Offset (~)

A path terminating with a leading tilde (`~`) followed by a number indicates a temporal position for audio or video measured in seconds.

› 3.1.6 Spatial Offset (@)

A path terminating with a leading at sign (`@`) followed by two colon-separated numbers indicates a 2D spatial position within an image or video. The two numbers represent scaled locations in the `x` and `y` axes, and MUST be in the range `0` to `100` regardless of the image's native or display dimensions (i.e., the upper left is `0:0` and the lower right is `100:100`).

› 3.1.7 Temporal-Spatial Offset (~ + @)

A temporal and a spatial position MAY be used together. In this case, the temporal specification MUST precede the spatial one syntactically (e.g., `~23.5@5.75:97.6` refers to a point 23.5 seconds into a video in the lower left of the frame).

› 3.1.8 Text Location Assertion ([)

An EPUB CFI MAY specify a substring that is expected to precede and/or follow the encountered point, but such assertions MUST occur only after a character offset.

For example, the following expression asserts that `yyy` is expected immediately before the encountered point using the sample content below:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3[yyy])
```

An additional substring that follows the encountered point can be given after a comma. For example:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/1:3[xx,y])
```

refers to the position marked by the asterisk:

```
x x x y y y 0 1 2 3 4 5 6 7 8 9
| | | * | | | | | | | | | | | |
```

If there is no preceding text, or only trailing text is specified, a comma MUST immediately precede the text assertion:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3[,y])
```

There is no restriction on the amount of the preceding and following text that can be included in the match. Text is taken from the document ignoring element boundaries and white space is always collapsed (i.e., a non-empty sequence of contiguous white space characters is always replaced with a single space character).

A Reading System can determine that the location referenced by the CFI is not the original intended location (due to non-matching text), and can use the preceding/trailing text to compute the set of steps that reach the desired destination in the content (see Intended Target Location Correction). The cost of this added robustness is that comparison (and sorting) of CFI strings can be performed only after logically stripping all bracketed substrings (see Sorting Rules).

› 3.1.9 Side Bias ([+ ;s=)

In some situations, it is important to preserve which side of a location a reference points to. For example, when resolving a location in a dynamically paginated environment, it would make a difference if a location is attached to the content before or after it (e.g., to determine whether to display the verso or recto side at a page break).

The s parameter is used to preserve this sided-ness aspect of a location. It can take two values: 'b' ("before") means that the location is attached to the content that precedes (according to the XML serialization document order), 'a' ("after") refers to the content that follows. This parameter MUST always be used inside square brackets at the end of the CFI, even if the ID [XML] or text location assertion is empty.

The location just after yyy in the sample content below can be expressed as belonging with the content before it as follows:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3[;s=b])
```

Equally, it can be expressed including a text location assertion as:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3[yyy;s=b])
```

The location at the start of em element can be attached to the content preceding the em element as follows:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2[;s=b])
```

If the side bias in the preceding example was set to a rather than b, the location would be attached to the child content of the em element, not the content following the em element.

Since side bias is expressed as a parameter, it does not participate in CFI comparison (see Sorting Rules).

Side is not defined for locations with spatial offset.

NOTE

Side bias is only meaningful when some type of break falls at the location (e.g., a page break or line break).

› 3.1.10 Examples

This section is informative

Given the following Package Document:

```
<?xml version="1.0"?>

<package version="2.0"
         unique-identifier="bookid"
         xmlns="http://www.idpf.org/2007/opf"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:opf="http://www.idpf.org/2007/opf">

    <metadata>
        <dc:title>…</dc:title>
        <dc:identifier id="bookid">…</dc:identifier>
        <dc:creator>…</dc:creator>
        <dc:language>en</dc:language>
    </metadata>

    <manifest>
        <item id="toc"
              properties="nav"
              href="toc.xhtml"
              media-type="application/xhtml+xml"/>
        <item id="titlepage"
              href="titlepage.xhtml"
              media-type="application/xhtml+xml"/>
        <item id="chapter01"
              href="chapter01.xhtml"
              media-type="application/xhtml+xml"/>
        <item id="chapter02"
              href="chapter02.xhtml"
              media-type="application/xhtml+xml"/>
        <item id="chapter03"
              href="chapter03.xhtml"
              media-type="application/xhtml+xml"/>
        <item id="chapter04"
              href="chapter04.xhtml"
              media-type="application/xhtml+xml"/>
    </manifest>

    <spine>
        <itemref id="titleref"  idref="titlepage"/>
        <itemref id="chap01ref" idref="chapter01"/>
        <itemref id="chap02ref" idref="chapter02"/>
        <itemref id="chap03ref" idref="chapter03"/>
        <itemref id="chap04ref" idref="chapter04"/>
    </spine>

</package>
```

and the XHTML Content Document `chapter01.xhtml`:

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>…</title>
    </head>

    <body id="body01">
        <p>…</p>
        <p>…</p>
        <p>…</p>
        <p>…</p>
        <p id="para05">xxx<em>yyy</em>0123456789</p>
        <p>…</p>
        <p>…</p>
        <img id="svgimg" src="foo.svg" alt="…"/>
        <p>…</p>
        <p>…</p>
    </body>
</html>
```

Then the EPUB CFI:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/3:10)
```

refers to the position right after the digit `9` in the paragraph with the ID `para05`. When producing CFIs for text locations, unless the text is defined by an `img` element's `alt` tag, one SHOULD always start with the reference to the (possibly-empty) chunk of XML character data that corresponds to the location and then trace the ancestor and reference chain to the Package Document root.

The following examples show how EPUB CFIs can be constructed to reference additional content locations.

Reference to the `img` element.

```
epubcfi(/6/4[chap01ref]!/4[body01]/16[svgimg])
```

Reference to the location just before `xxx`.

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/1:0)
```

Reference to the location just before `yyy`.

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:0)
```

Reference to the location just after `yyy`.

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3)
```

› 3.2 Sorting Rules

In order to sort or compute relative locations of multiple EPUB CFIs referencing the same EPUB Publication, the following rules MUST be applied:

1. The EPUB CFI scheme data MUST be in unescaped form, as per the rules described in Character Escaping.
2. all bracketed assertions are removed (ignored) entirely;
3. steps that come earlier in the sequence are more important;
4. XML elements, references to chunks of XML character data, character offsets and temporal positions are sorted in natural order;
5. the `y` position is more important than `x`;
6. omitted spatial position precedes all other spatial positions;
7. omitted temporal position precedes all other temporal positions;
8. temporal position is more important than spatial;
9. different step types come in the following order from least important to most important: character offset (`:`), child (`/`), temporal-spatial (`~` or `@`), reference/indirect (`!`).

› 3.3 Intra-Publication CFIs

An EPUB CFI can be used to reference content inside the container. This kind of referencing can be achieved by specifying a reference to the Package Document followed by a CFI, which MUST be resolved starting from the root `package` element.

For example, using the Package Document in the previous example, a reference to the last location in `chapter01.xhtml` might be written as follows:

```
../pub.opf#epubcfi(/6/4[chap01ref]!/4[body01]/10[para05]/2/1:3[;s=b])
```

› 3.4 Simple Ranges

EPUB CFIs allow the expression of simple ranges extending from a start location to an end location. A range MUST be expressed as a triple of *parent* path (`P`), *start* subpath (`S`) and *end* subpath (`E`), or of the form:

```
epubcfi(P,S,E)
```

The parent path MUST not be empty, and MUST end at a step that is common for resolving both the path of the start and end locations of the range, and each start and end subpath MUST resolve to a location in non-decreasing order in the document.

To determine the start and end locations of the range, the start and end subpaths MUST be concatenated to the parent path to create the start location path (`PS`) and end location path

(`PE`). The parent path SHOULD include the deepest possible common path leading to both the start and end path (in other words, the start and end location SHOULD NOT contain a common path). The start location MAY be empty, to avoid repetition of a common path in cases where the end location is situated within the subtree rooted at the start location.

Using the sample documents above, the following range would represents the text from the second `y` in `yyy` up to (and including) digit `3`:

```
epubcfi(/6/4[chap01ref]!/4[body01]/10[para05],/2/1:1,/3:4)
```

Ranges MUST be compared according to their `PS`, then `PE`, components. The start and end locations SHOULD reference points in the document that have the same "nature", that is to say elements and character offsets (document structure), temporal offsets (timed media), spatial offsets (visual media), or the combined temporal-spatial offsets. In the case of temporal offsets, the start and end locations SHOULD reference the same timed media. In the case of spatial offsets, the start and end locations SHOULD reference the same visual media. This specification does not define expected behaviors, such as how the combination of two spatial offsets (i.e., start and end locations within a visual media) is to be interpreted by processing agents, including production tools and reading systems.

It is not valid to use a path to an element as a shorthand for the range from the beginning to the end of the element. Single path notation always denotes a location point, and range is represented by the notation described above. There is no special step to produce a reference to the end of an element, as that would make sorting impossible without consulting the content of the document.

If range is used where single location is expected by the context, the start location MUST be used.

Side-bias parameters MUST NOT be used for ranges; the start of a range is implicitly attached to the content after the start location and the end is implicitly attached to the content before the end location.

› 3.5 Intended Target Location Correction

As an EPUB Publication can be updated, corrected or otherwise altered over time, it is useful to be able to derive an EPUB CFI for the modified document from one that targeted a previous version. This specification provides two mechanisms to detect and adapt to content changes that impact CFIs: IDs [XML] and text location assertions.

When a Reading System is processing a CFI, it SHOULD check the correctness of any encountered assertions. For example, given the path `/6/4[chap01ref]!…`, the Reading System SHOULD verify that the element has the ID matching `chap01ref` when processing element `4` (for this example, an `itemref` in the `spine`). If not, the Reading System SHOULD locate the ID `chap01ref` within the document and correct the CFI (e.g., if a new `itemref` was inserted before the `chap01ref itemref`, the desired element number would now be `6` and the corrected CFI would be `/6/6[chap01ref]!…`). Likewise, text location assertions SHOULD be used to check referenced target locations, and used to derive a corrected CFI that targets the desired text location.

If one of the assertions fails during processing, and a corrected CFI can not be derived (the ID is not found in the document, or text matches could not be found), the CFI MUST be considered an invalid reference. In cases where a Reading System cannot check for correctness (e.g., document-resident XML IDs are not available at CFI processing time), a Reading System MUST ignore the CFI assertions.

This notion of correcting CFIs can lead to circumstances where two different CFIs point to the same location (i.e., the "stale" CFI, pre-correction, and the corrected CFI). The corrected CFI SHOULD be used where possible. A Reading System and any surrounding content management system SHOULD attempt to replace stale CFIs with their corrected versions where possible.

> NOTE
>
> This specification encourages the development of custom functions to assist with CFI correction where the intrinsic functionality is insufficient. Refer to *Extending EPUB CFIs* for more information on how to develop such functionality.

› 4 Extending EPUB CFIs

The provision for extensions (CSV parameter lists, prefixed by a parameter name, and separated by semicolons) allow Reading Systems to apply new or experimental heuristics to assist, for example, in migrating EPUB CFI fragments to updated documents.

It is RECOMMENDED that any vendor-specific parameter names start with `vnd.` followed by the vendor name.

Implementations MUST ignore all parameters that they do not understand or cannot parse.

› References

Normative References

[EPUB 3.1] *EPUB 3.1* .

[EPUB-CFI] *EPUB Canonical Fragment Identifier (epubcfi) Specification* .

[HTML] *HTML* .

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

[RFC2279] *UTF-8, a transformation format of ISO 10646* *(RFC 2279)* . F. Yergeau, et al. January 1998.

[RFC2396] *Uniform Resource Identifiers (URI): Generic Syntax* *(RFC 2396)* . T. Berners-Lee, et al. August 1998.

[RFC2732] *Format for Literal IPv6 Addresses in URL's* *(RFC 2732)* . R. Hinden, et al. December 1999.

[RFC3986] *Uniform Resource Identifier (URI): Generic Syntax (RFC 3986)* . Berners-Lee, et al. January 2005.

[RFC3987] *Internationalized Resource Identifiers (IRIs) (RFC 3987)* . M Duerst, et al. January 2005.

[SVG] *Scalable Vector Graphics (SVG)* .

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

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

Informative References

[DOM2 Traversal Range] *Document Object Model (DOM) Level 2 Traversal and Range Specification*. Joe Kesselman, et al. 13 November, 2000.

[ECMA-262] *ECMA-262 6th Edition, ECMAScript® 2015 Language Specification*. June 2015.

[FragIDBestPractices] *Best Practices for Fragment Identifiers and Media Type Definitions*.

[XPTRSH] *XPointer Shorthand Notation* . 25 March 2003.

ICS 35.240.30

Price based on 20 pages