TECHNICAL SPECIFICATION

ISO/IEC TS 30135-6

First edition
2014-11-15

Information technology — Digital publishing — EPUB3 —

Part 6: EPUB Canonical Fragment Identifier

Technologies de l'information — Publications numériques — EPUB3 —

Partie 6: Identificateurs de fragment canoniques EPUB

Reference number
ISO/IEC TS 30135-6:2014(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web 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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

In other circumstances, particularly when there is an urgent market requirement for such documents, the joint technical committee may decide to publish an ISO/IEC Technical Specification (ISO/IEC TS), which represents an agreement between the members of the joint technical committee and is accepted for publication if it is approved by 2/3 of the members of the committee casting a vote.

An ISO/IEC TS is reviewed after three years in order to decide whether it will be confirmed for a further three years, revised to become an International Standard, or withdrawn. If the ISO/IEC TS is confirmed, it is reviewed again after a further three years, at which time it must either be transformed into an International Standard or be withdrawn.

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.

ISO/IEC TS 30135 series were prepared by Korean Agency for Technology and Standards (as KS X 6070 series) with International Digital Publishing Forum and were adopted, under a special “fast-track procedure”, by Joint Technical Committee ISO/IEC JTC 1, Information technology, in parallel with its approval by the national bodies of ISO and IEC.

ISO/IEC TS 30135 consists of the following parts, under the general title *Information technology — Document description and processing languages — EPUB 3:*

- *Part 1: Overview*
- *Part 2: Publications*
- *Part 3: Content Documents*
- *Part 4: Open Container Format*
- *Part 5: Media Overlay*
- *Part 6: Canonical Fragment Identifier*
- *Part 7: Fixed-Layout Documents*

EPUB Canonical Fragment Identifier (epubcfi) Specification

Recommended Specification 11 October 2011

This version

http://www.idpf.org/epub/linking/cfi/epub-cfi-20111011.html

Latest version

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

Previous version

http://www.idpf.org/epub/linking/cfi/epub-cfi-20110908.html

A diff of changes from the previous draft is available at this link.

Please refer to the errata for this document, which may include some normative corrections.

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

Table of Contents

1 Overview

1.1 Purpose and Scope

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 a 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 required 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.

› 1.2 Terminology

Please refer to the EPUB Specifications 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 Publication.

Intra-Publication EPUB CFI

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

Refer to Intra-Publication CFIs for more information.

› 1.3 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.

› 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 of end 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 Publication (`demo.epub`). Using the processing rules defined in Path Resolution, any Reading System can parse this syntax, open the corresponding Content Document in the 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 may be defined in the future for all XML+ZIP-based file formats.

› 2.2 Syntax

(EBNF productions ISO/IEC 14977)

fragment	=	"epubcfi(" , (path | range) , ")" ;
path	=	step , local_path ;
range	=	path , "," , local_path , "," , local_path ;
local_path	=	{ step | "!" } , [termstep] ;
step	=	"/" , integer , ["[" , assertion , "]"] ;
termstep	=	terminus , ["[" , assertion , "]"] ;
terminus	=	(":" , integer) | ("@" , number , ":" , number) | ("~" , number) | ("~" , number , "@" , number , ":" , number) ;
number	=	(digit-non-zero , { digit } , ["." , { digit } , digit-non-zero]) | (zero , "." , { digit } , digit-non-zero) ;
integer	=	zero | (digit-non-zero , { digit }) ;
assertion	=	[csv] , { 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 } ;
	=	

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

› **Unicode Characters**

The definition of allowed Unicode characters is the same as [XML 1.0]. 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 can either be navigational or terminating. Navigational steps may be repeated as necessary (e.g., to count elements, to process children or to follow references). There may be only one terminating step, which, if present, must be the last step in the sequence.

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 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(/6/7[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* used 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 required 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 may 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 XHTML 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 -> XHTML] becomes [XHTML -> 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 opening square brackets as well as the escaped circumflex character itself (the unescape text is 'Φ-"spa ce"-99%-aa[bb]^'):

```
epubcfi(/6/7!/4/10/2/1:3[Φ-"spa ce"-99%-aa^[bb^]^^])
```

When taking part in a IRI, the space character within the assertion may 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 copying all the text after the '#' character). The circumflex '^' also falls within a 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:

```
book.epub#epubcfi(/6/7!/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 '"'. 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:

```
book.epub#epubcfi(/6/7!/4/10/2/1:3[Φ-&#x22;spa%20ce-99%25&#x22;-
aa^[bb^]^^])
```

Should 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:

```
book.epub#epubcfi(/6/7!/4/10/2/1:3[%d0%a4-&#x22;spa%20ce&#x22;-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) :

```
book.epub#epubcfi(/6/7!/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 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 Node (/)

A step with a slash (`/`) followed by an integer refers to a child node or nodes in the following manner:

- Each element is assigned an *even* positive index: the first element is given index `2`, the second element index `4`, etc.
- Each (possibly empty) collection of non-element nodes before the first element, between elements, and after the last element are given *odd* indices according to their position (these typically refer to the text of the Publication).
- Non-element nodes that are not text nodes are always ignored (for the purposes of this specification, a text node includes text, CDATA sections and entity references).

This indexing method ensures that node identification is not sensitive to XML parser handling of whitespace text nodes, CDATA sections and entity references (e.g., to avoid the ambiguity that can arise depending on whether a parser collapses whitespace-only text nodes, keeps text, CDATA sections and entity references as distinct nodes or doesn't, or breaks text in multiple nodes).

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 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 may determine that the location referenced by the CFI is not the original intended location, and may 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 may be performed only after logically stripping all bracketed substrings (see Sorting Rules).

› 3.1.3 Step Indirection (!)

A step with a leading exclamation point (!) indicates that the reference must be followed and the next step applied starting from the new target node (or root element node when a complete XML document is referenced).

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 [HTML5] `iframe` and `embed` elements, references are defined by the `src` attribute
- For the [HTML5] `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 [HTML5] (or [SVG]) `a` element.

› 3.1.4 Terminating Step – Character Offset (:)

A terminating step with a leading colon (:) followed by an integer refers to a character offset. The given character offset may apply to an element node only if this element is the [HTML5] `img` element with an `alt` attribute containing the text to which the character offset applies.

For text nodes, 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.

A character offset terminating step may be present only following 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 text in a text node.

No other steps may follow a character offset terminating step.

› 3.1.5 Terminating Step – Temporal Offset (~)

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

No other steps can follow a temporal offset terminating step.

› 3.1.6 Terminating Step – Spatial Offset (@)

A terminating step 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`).

No other steps can follow a spatial offset terminating step.

› 3.1.7 Terminating Step – 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).

No other steps can follow a temporal-spatial position terminating step.

› 3.1.8 Text Location Assertion ([)

An EPUB CFI may specify a substring that should precede and/or follow the encountered point, but such assertions must occur only after a character offset terminating step.

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 whitespace is always collapsed (i.e., a non-empty sequence of contiguous whitespace characters is always replaced with a single space character).

A Reading System may determine that the location referenced by the CFI is not the original intended location (due to non-matching text), and may 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 may 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` means that the location belongs with the content *before* it and `a` with the content *after*. 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 terminus.

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 text node or text node collection (even if empty) 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 child nodes, 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