Table of contents

1 Chunking....................................................................................................................................................3
  1.1 About the @chunk attribute............................................................................................................... 3
  1.2 Processing chunk="combine"............................................................................................................ 4
  1.3 Processing chunk="split"................................................................................................................... 4
  1.4 Using the @chunk attribute for other purposes................................................................................. 4
  1.5 Examples of the @chunk attribute.....................................................................................................4
    1.5.1 Example: Using @chunk to combine all documents into one................................................... 5
    1.5.2 Example: Using @chunk to render a single document from one branch............................... 6
    1.5.3 Example: Using @chunk to combine a group of topics............................................................... 8
    1.5.4 Example: Using @chunk to combine nested documents.......................................................... 9
    1.5.5 Example: Using @chunk to split documents........................................................................... 10
    1.5.6 Example: Using @chunk to split nested documents............................................................... 13
    1.5.7 Example: When @chunk is ignored........................................................................................ 15
    1.5.8 Example: Combining topics within a split context.................................................................16
    1.5.9 Example: Managing links when chunking................................................................................16
  A Aggregated RFC-2119 statements..........................................................................................................19
1 Chunking

Content often needs to be delivered in a different granularity than it is authored. The @chunk attribute enables map authors to specify that multiple source documents are combined into a single document for delivery, or that a single source document is split into multiple documents for delivery.

1.1 About the @chunk attribute

The @chunk attribute is designed to handle cases where the best organization for authoring DITA topics is not equivalent to the best organization for publishing those topics.

The @chunk attribute is composed of a single token without any white space. DITA defines two tokens for the @chunk attribute: combine and split. Other tokens can be defined by applications, but support for those tokens will vary.

**chunk="combine"**

The "combine" token in @chunk is intended for cases where a publishing process normally results in a single output artifact for each source XML document. When some or all of those source XML documents are better presented as a single output artifact, setting chunk="combine" instructs a processor to combine the referenced source documents for rendering purposes.

**chunk="split"**

The "split" token in @chunk is intended for cases where a publishing process normally results in a single output artifact for each single source XML document, regardless of how many DITA topics exist within each source document. When a source XML document containing many topics is better rendered as multiple output artifacts, setting chunk="split" instructs a processor to split each topic from the referenced source document into its own document for rendering purposes.

The @chunk attribute describes how a processor can split or combine source DITA documents into alternate organizational schemes for rendering purposes. This means that the @chunk attribute is only relevant when the organization of source DITA documents has an effect on organization of published documents.

The @chunk attribute does not cascade.

The following rules apply to all values of the @chunk attribute:

001 (19)

- When the source document organization has no effect on published output, such as when producing a single PDF or EPUB, processors **MAY** ignore the @chunk attribute.
- When the @chunk attribute results in more or fewer documents based on the combine or split tokens, the hierarchy of topics within the resulting map and topic organization **SHOULD** match the hierarchy in the original topics and maps.
- When the @chunk attribute results in more or fewer documents, processors **MAY** create their own naming schemes for those reorganized documents.
- @chunk attribute values apply to DITA topic documents referenced from a map. Processors **MAY** apply equivalent processing to non-DITA documents.
1.2 Processing chunk="combine"

Setting chunk="combine" instructs a processor to combine the reference source documents for rendering purposes. A single result document is generated.

The following rules apply:

- When specified on the root element of a map, all source DITA documents referenced by the map are treated as one DITA document.
- When specified on a branch of a map, all source DITA documents referenced within that branch are treated as one DITA document.

  Note This is true regardless of whether the element that specifies @chunk refers to a topic or specifies a heading. In cases such as <topicgroup> where a grouping element specifies chunk="combine", the equivalent DITA document would be a single DITA document with a root element grouping peer topics.

- When specified on a reference to a map, all source DITA documents within the scope of the referenced map are treated as one DITA document.
- Once chunk="combine" is specified on a map, branch, or map reference, all source DITA documents grouped by that reference are treated as a single resource. Any additional @chunk attributes on elements within the hierarchy are ignored.

1.3 Processing chunk="split"

Setting chunk="split" instructs a processor to split each topic from the referenced source document into its own document for rendering purposes. Multiple result documents are generated.

The following rules apply:

- When specified on a <topicref> element that refers to a source DITA document, it indicates that all topics within the referenced document should be rendered as individual documents.
- When specified on an element such as <topicgroup> that does not refer to a topic or result in a published topic, the attribute has no meaning.
- When specified on the root element of a map, chunk="split" sets a default operation for all source DITA documents in the map (outside the context of relationship tables). The default split value is used except where a combine value is encountered, in which case combine takes over for that entire branch.

1.4 Using the @chunk attribute for other purposes

Applications can define additional tokens for use in the @chunk attribute. These tokens are necessarily implementation dependent and might not be supported by other applications.

1.5 Examples of the @chunk attribute

These examples illustrate the processing expectations for various scenarios that involve the @chunk attribute. The processing examples use either before and after sample markup or expanded syntax that shows the equivalent markup without the @chunk attribute.

  Note Examples use sample files with modified file names to help illustrate equivalent before and after resolution of @chunk attributes. However, there is no requirement for implementations processing @chunk to generate files, as long as the rendered result is split or combined as described. If generating files, actual file names are implementation dependent.
1.5.1 Example: Using @chunk to combine all documents into one

Where a publishing system typically would render each topic document as an independent result document, the @chunk attribute can be used to render all content as a single document.

Consider the following source documents, with root map input.ditamap:

**Figure 1: Input map without chunking**

```
input.ditamap:
<map>
  <title>Lesson plan</title>
  <topicref href="background.dita">
    <!-- more background topics -->
  </topicref>
  <topicref href="goals.dita">
    <!-- more goal topics -->
  </topicref>
  <!-- more topics -->
</map>
```

```
background.dita:
<topic id="background">
  <title>Prerequisite concepts</title>
  <shortdesc>This information is necessary before starting</shortdesc>
  <body><!-- ...background content... --></body>
</topic>
```

```
goals.dita:
<topic id="goals">
  <title>Lesson goals</title>
  <shortdesc>After you complete the lesson, ...</shortdesc>
  <body><!-- ...goal content... --></body>
</topic>
```

For many systems or output formats, each document in the map is rendered as an independent document. For example, rendering this map as HTML5 might result in background.html and goals.html (along with other HTML5 files). In such cases, if output requirements demand only a single result document, adding chunk="combine" to the root map element instructs a processor to render one document that combines all topics.

**Figure 2: Root map with chunking specified**

```
input.ditamap:
<map chunk="combine">
  <title>Lesson plan</title>
  <topicref href="background.dita">
    <!-- more background topics -->
  </topicref>
  <topicref href="goals.dita">
    <!-- more goal topics -->
  </topicref>
  <!-- more topics -->
</map>
```

The result of evaluating this @chunk attribute is equivalent to the following map and topic document; content from all topics within the map is combined into a single result, with a topic order and topic nesting structure that match the original map hierarchy.

**Figure 3: Equivalent source content**

```
input.ditamap:
<map>
  <title>Lesson plan</title>
  <topicref href="input.dita"/>
```

1.5.2 Example: Using @chunk to render a single document from one branch

Where a publishing system typically would render each topic document as an independent result document, the @chunk attribute can be used to render individual branches of a map as single documents.

Consider the following source documents, with root map input.ditamap:

**Figure 4: Input map without chunking**

```
input.ditamap:
<map>
  <title>Lesson plan</title>
  <topicref href="goals.dita">
    <!-- more goal topics -->
  </topicref>
  <topicref href="firstLesson.dita">
    <!-- more tasks in the first lesson -->
  </topicref>
  <topicref href="nextLesson.dita">
    <!-- more tasks in the next lesson -->
  </topicref>
  <!-- More branches -->
</map>
firstLesson.dita:
<task id="firstLesson">
  <title>Starting to work with scissors</title>
  <shortdesc>This lesson will teach ...</shortdesc>
  <taskbody><!-- ... --></taskbody>
</task>
nextLesson.dita:
<task id="nextLesson">
  <title>Advanced cutting</title>
  <shortdesc>This lesson will introduce complicated shapes...</shortdesc>
  <taskbody><!-- ... --></taskbody>
</task>
```

For many systems or output formats, each document in the map is rendered as an independent document. For example, rendering this map as HTML5 might result in goals.html, firstLesson.html, and nextLesson.html, while child documents within each branch would each result in their own HTML files.

When output requirements demand that portions of the map be combined into a single document, adding chunk="combine" to a branch of the map instructs a processor to render one document that combines
all topics in that branch. This is particularly useful when the topics need to be rendered independently for other contexts, or when the way topics are contributed makes creating a single source document impossible.

In the following sample, the original map is updated with @chunk attributes to indicate that each lesson branch is rendered as a single result document; topics in the first branch with goals.dita are not affected as a result of the @chunk attribute.

**Figure 5: Map with chunking specified for one branch**

```xml
input.ditamap:
<map>
  <title>Lesson plan</title>
  <topicref href="goals.dita">
    <!-- more goal topics -->
  </topicref>
  <topicref href="firstLesson.dita" chunk="combine">
    <!-- more tasks in the first lesson -->
  </topicref>
  <topicref href="nextLesson.dita" chunk="combine">
    <!-- more tasks in the next lesson -->
  </topicref>
  <!-- More branches -->
</map>
```

The result of evaluating this @chunk attribute is equivalent to the following map and topic documents. Content from each branch where @chunk attribute is set is combined into a single result document, with an order and topic nesting structure that matches the original map hierarchy. Content from outside of those branches remains unchanged.

**Figure 6: Equivalent source content**

```xml
input.ditamap:
<map>
  <title>Lesson plan</title>
  <topicref href="goals.dita">
    <!-- more goal topics -->
  </topicref>
  <topicref href="firstLesson.dita"/>
  <topicref href="nextLesson.dita"/>
  <!-- More branches -->
</map>

firstLesson.dita:
<task id="firstLesson">
  <title>Starting to work with scissors</title>
  <shortdesc>This lesson will teach ...</shortdesc>
  <taskbody><!-- ... -->
  <!-- more tasks in the first lesson -->
</task>

nextLesson.dita:
<task id="nextLesson">
  <title>Advanced cutting</title>
  <shortdesc>This lesson will introduce complicated shapes...</shortdesc>
  <taskbody><!-- ... -->
  <!-- more tasks in the next lesson -->
</task>
```
1.5.3 Example: Using @chunk to combine a group of topics

The @chunk attribute can be used on grouping elements to combine multiple source documents into one result document.

Assume the following map input.ditamap, where @chunk is used on both <topicgroup> and <topichead>.

Figure 7: Input map

```xml
<map>
  <title>Groups are combined</title>
  <topicgroup chunk="combine">
    <topicref href="ingroup1.dita"/>
    <topicref href="ingroup2.dita"/>
  </topicgroup>
  <topichead chunk="combine">
    <topicmeta>
      <navtitle>Heading for a branch</navtitle>
    </topicmeta>
    <topicref href="inhead1.dita"/>
    <topicref href="inhead2.dita"/>
  </topichead>
</map>
```

The result of evaluating the @chunk attribute on <topicgroup> is equivalent to a single DITA document with the content of both ingroup1.dita and ingroup2.dita.

The @chunk attribute on <topichead> also results in a single DITA document. In many applications, a <topichead> is equivalent to a single title-only topic; in that case, the chunked result is equivalent to a root topic with the title “Heading for a branch”, containing as child topics the content of both inhead1.dita and inhead2.dita. If <topichead> is ignorable in the current processing context, the chunked result would be equivalent to processing <topicgroup> (a single DITA document with the content of both inhead1.dita and inhead2.dita).

Figure 8: Equivalent source content

```xml
<map>
  <title>Groups are combined</title>
  <topicref href="chunkgroup-1.dita"/>
  <topicref href="chunkgroup-2.dita"/>
</map>
```

chunkgroup-1.dita
```xml
<dita>
  <!-- content of ingroup1.dita -->
  <!-- content of ingroup2.dita -->
</dita>
```

chunkgroup-2.dita
```xml
<dita>
  <topic id="head">
    <title>Heading for a branch</title>
    <!-- content of inhead1.dita -->
    <!-- content of inhead2.dita -->
  </topic>
</dita>
```
1.5.4 Example: Using @chunk to combine nested documents

Special attention is necessary when combining a nested map hierarchy that includes documents with their own nested topics.

Consider the following source map input.ditamap:

Figure 9: Input map without chunking

```xml
<input.ditamap>
  <map chunk="combine">
    <title>Generation example</title>
    <topicref href="ancestor.dita">
      <topicref href="middle.dita">
        <topicref href="child.dita"/>
      </topicref>
    </topicref>
  </map>
</input.ditamap>
```

In this case, the @chunk attribute instructs a processor to treat the three topics as a single combined DITA document, while preserving the original map hierarchy. Now consider the following three topic documents, each of which includes nested or peer topics:

Figure 10: Source documents with nested structures

```xml
<ancestor.dita>
  <dita>
    <topic id="ancestor-first">
      <title>First major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
    <!-- more topics in ancestor composite doc -->
    <topic id="ancestor-last">
      <title>Last major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
    <topic id="ancestor-last-child">
      <title>Child of last major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
  </dita>
</ancestor.dita>

<middle.dita>
  <topic id="middle-root">
    <title>Root topic in middle doc</title>
    <body> <!-- body content, maybe more children of middle topic's root -->
  </topic>
</middle.dita>

<child.dita>
  <topic id="child">
    <title>Small child topic</title>
    <!-- small child topic content -->
  </topic>
</child.dita>
```

When chunk="combine" is evaluated, the three source documents are combined into one. Both the ancestor and middle documents have child topics that need to be taken into account.

- ancestor.dita has a root <dita> element, so content from each nested topic reference is located after any nested topics within the final child of the <dita> element.
middle.dita does not have `<dita>` but does have a nested topic, so content from any nested topic references is located after that nested topic.

**Figure 11: Equivalent source content**

```xml
input.ditmap:
  <map>
    <title>Generation example</title>
    <topicref href="input.dita"/>
  </map>

input.dita:
  <dita>
    <topic id="ancestor-first">
      <title>First major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
    <!-- more topics in ancestor composite doc -->
    <topic id="ancestor-last">
      <title>Last major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
    <topic id="ancestor-last-child">
      <title>Child of last major topic in ancestor composite doc</title>
      <!-- ...topic content... -->
    </topic>
    <!-- content of middle.dita combined here -->
    <topic id="middle-root">
      <title>Root topic in middle doc</title>
      <body><!-- ... --></body>
      <topic id="middle-child">
        <title>Child of root topic in middle doc</title>
        <!-- body content, maybe more children of middle topic's root -->
      </topic>
      <!-- content of child.dita combined here -->
      <topic id="child">
        <title>Small child topic</title>
        <!-- small child topic content -->
      </topic>
  </dita>
```

### 1.5.5 Example: Using `@chunk` to split documents

When topics are most easily created or generated in a single DITA document, `chunk="split"` will instruct processors to render them individually when possible.

**Splitting a single document in the map**

Consider the following example, where a map includes generated topics used to document message numbers from an application:

**Figure 12: Source map and topics**

```xml
<map>
  <title>Message guide for WidgetAnalyzer</title>
  <topicref href="about.dita">
    <topicref href="messages-install.dita"/>
    <topicref href="messages-run.dita"/>
    <topicref href="messages-other.dita"/>
  </topicref>
</map>

about.dita:
  <topic id="about">
    <title>About this guide</title>
    <shortdesc>Warnings or errors will appear if...<shortdesc>
```
In a normal build to HTML5, this map might result in four result documents about.html, messages-install.html, messages-run.html, and messages-other.html. With hundreds of messages in messages-run.dita, it might be better in some situations to render one result document for each message topic in the document. This can be done by setting chunk="split" on the topic reference.

Figure 13: Splitting all topics in one document

The result of evaluating @chunk in this case is equivalent to the following map and topics. While messages-run.dita now is split into hundreds of topics, other topics in the map are unaffected.

Figure 14: Equivalent source content
RUN001.dita:
<topic id="RUN001">
<title>RUN001: Failed to initialize</title>
<!-- explanation and recovery... -->
</topic>

RUN999.dita:
<topic id="RUN999">
<title>RUN999: Out of memory</title>
<!-- explanation and recovery... -->
</topic>

Note  Because the @chunk attribute does not cascade, even if the reference to messages-install.dita had child topic references, they would be unaffected by the chunk="split" value in this example.

Splitting every document in the map

Similarly, because setting chunk="split" on the map element sets a default for the entire map, the following change to the original map would result in every referenced DITA document being split into one document per topic. The only source document not affected by this split is about.dita, because it only contained a single topic to begin with.

Figure 15: Splitting every topic in the map

Using chunk="split" on the map is equivalent to the following structure:

- about.dita is unchanged.
- messages-install.dita is split into one document per message (as in the previous example that split messages-run.dita).
- messages-run.dita is split exactly as in the previous example.
- messages-other.dita contains a root topic and two child topics, so it results in three documents. The hierarchy of those documents is preserved in the map.

Figure 16: Equivalent source content
1.5.6 Example: Using @chunk to split nested documents

Special attention is necessary when evaluating the map hierarchy that results from splitting a documents with their own nested topics.

Consider the following source map `input.ditamap`:

**Figure 17: Input map without chunking**

```
input.ditamap:
<map chunk="split">
  <title>Generation example</title>
  <topicref href="ancestor.dita">
    <topicref href="middle.dita">
      <topicref href="child.dita"/>
    </topicref>
  </topicref>
</map>
```

In this case, the `@chunk` attribute instructs a processor to render every topic in each of the three documents as its own document, while preserving any hierarchy from those documents. Now consider the following three topic documents, each of which includes nested or peer topics:

**Figure 18: Source documents with nested structures**

```
ancestor.dita:
<dita>
  <topic id="ancestor-first">
    <title>First major topic in ancestor composite doc</title>
    <!-- ...topic content... -->
  </topic>

  <!-- more topics in ancestor composite doc -->
  <topic id="ancestor-last">
    <title>Last major topic in ancestor composite doc</title>
    <!-- ...topic content... -->
  </topic>

  <title>Child of last major topic in ancestor composite doc</title>
</dita>
```

```
When chunk="split" is evaluated, both ancestor.dita and middle.dita are split and treated as multiple DITA topic documents. child.dita is only a single topic and has nothing to split.

- ancestor.dita has a root <dita> element, so it results in multiple peer topic references (or branches) in the map. Topic references nested within the original reference to ancestor.dita are now located within the reference to "ancestor-last" (the last topic child of the <dita> element).
- middle.dita has nested topics, so results in its own new hierarchy within the map. Content from the nested topic reference is now located within the reference to the root topic from middle.dita, but after any references to child topics.

Figure 19: Equivalent source content
1.5.7 Example: When @chunk is ignored

The @chunk attribute is ignored in some cases, such as when chunk="combine" is already in effect or when chunk="split" is specified on a grouping element.

Ignoring @chunk when already combining topics

In the following example, evaluating @chunk results in one rendered document for each branch of the map. Any additional @chunk values within that branch are ignored (including @chunk values within any referenced maps).

Figure 20: Chunk within a combined branch

```xml
<map>
  <title>Ignoring chunking when already combined</title>
  <topicref href="branchOne.dita" chunk="combine">
    <!-- @chunk ignored for branchOneChild.dita -->
  </topicref>
  <topicref href="branchTwo.dita" chunk="combine">
    <!-- Any @chunk within submap.ditamap is ignored -->
  </topicref>
  <topicref href="branchOneChild.dita" chunk="split"/>
  <topicref href="submap.ditamap" format="ditamap"/>
</map>
```

Ignoring @chunk on a grouping element

In the following example, chunk="split" is specified on two grouping elements.

Figure 21: Chunk within a combined branch

```xml
<map>
  <title>Trying to "split" groups</title>
  <topicgroup chunk="split">
    <topicref href="ingroup1.dita"><--...-->\</topicref>
    <topicref href="ingroup2.dita"><--...-->\</topicref>
  </topicgroup>
  <topichead chunk="split">
    <topicmeta><navtitle>Heading for a branch</navtitle></topicmeta>
    <topicref href="inhead1.dita"><--...-->\</topicref>
    <topicref href="inhead2.dita"><--...-->\</topicref>
  </topichead>
</map>
```

- The @chunk attribute on the <topicgroup> is ignored; it does not cascade, and there is no referenced topic, so it has no effect.
- In some cases, an implementation might treat the <topichead> element as equivalent to a single title-only topic, while in other cases it might be ignored. In either case the @chunk value has no effect. If the <topichead> is treated as a title-only topic, it cannot be split further; if it is ignored for the current processing context, it is no different than the <topicgroup>.
1.5.8 Example: Combining topics within a split context

While @chunk attributes are ignored when a "combine" action is already in effect, it is possible to use chunk="combine" when split is otherwise in effect.

Assume the following map, where chunk="split" on the root element means that all topic documents within this map structure are split by default, but a branch within the map sets chunk="combine".

**Figure 22: Map with default "split" action, that also uses "combine"**

```xml
<map chunk="split">
    <title>Split most, but not one branch</title>
    <topicref href="splitme.dita">...</topicref>
    <topicref href="exception.dita" chunk="combine">...</topicref>
    <topicref href="splitmetoo.dita">...</topicref>
</map>
```

Assume as well that no other @chunk attributes are specified in this map. The following points are true when @chunk is evaluated:

1. The document `splitme.dita` is treated as multiple split documents when it contains more than one topic. The same is true for any other document within that branch.
2. The second branch (beginning with `exception.dita`) is treated as a single DITA document, combining all topic documents within that branch.
3. The document `splitmetoo.dita` is treated as multiple split documents when it contains more than one topic. The same is true for any other document within that branch.

1.5.9 Example: Managing links when chunking

Link management with @chunk is often straightforward; in most cases where URI-based linking is ambiguous, using indirect links and @keyref will give the correct result.

**Input topics for following examples**

The following map and topics are used for all examples in this topic.

**Figure 23: input.ditamap**

```xml
<map>
    <title>Map with chunks and links</title>
    <keydef href="splitThis.dita" keys="splitThisKey"/>
    <keydef href="splitThis.dita#splitThisChild" keys="splitThisChildKey"/>
    <topicref href="splitThis.dita" chunk="split" keys="explicitSplitKey"/>
    <topicref href="combineThis.dita" keys="combineThisKey"/>
    <topicref href="combinedChild.dita" keys="combinedChildKey"/>
</map>
```

**Figure 24: Topics used by input.ditamap**

```xml
splitThis.dita:
    <topic id="splitThisRoot">
        <title>Root topic in split document</title>
        <!-- ... -->
    </topic>
    <topic id="splitThisChild">
        <title>Child topic in split document</title>
        <!-- ... -->
    </topic>
</splitThis.dita>
```
Topics that are rendered only once when publishing

Assume that the map above is a root map or is used by another map does not otherwise render the three topic documents. In that case, the following is true:

- `splitThis.dita` is rendered as two documents. For this example, assume a processor creates two documents with names taken from the topic ID, so that topic becomes `splitThisRoot.dita` and `splitThisChild.dita`.
- The branch with `combineThis.dita` is rendered as one document together with the content of `combinedChild.dita`. For this example, assume a processor merges the child topic into the file `combineThis.dita`.
- All links using `href="splitThis.dita"`, `keyref="splitThisKey"`, or `keyref="explicitSplitKey"` will resolve to `splitThisRoot.dita` (the only rendered instance of that topic).
- All links using `href="splitThis.dita#splitThisChild"` or `keyref="splitThisChildKey"` will resolve to `splitThisChild.dita` (the only rendered instance of that topic).
- All links using `href="combinedChild.dita"` or `keyref="combinedChildKey"` will resolve to that topic within `combineThis.dita` (the only rendered instance of that topic).

Topics that are rendered twice when publishing

Now assume that the map above is reused in another context that also renders all three topic documents as originally authored. As a result, each of the three documents in this map (`splitThis.dita`, `combineThis.dita`, and `combinedChild.dita`) are rendered more than once.

When each of these documents is rendered twice, the following is true:

- The original source document `splitThis.dita` is rendered twice. Based on the map above, assume a processor creates two documents with names taken from the topic ID, so that topic becomes `splitThisRoot.dita` and `splitThisChild.dita`. At the same time, `splitThis.dita` is rendered in another context as a single document, with a different name.
- Based on the map above, the branch that starts with the original source document `combineThis.dita` is rendered as one document combined with the content of `combinedChild.dita`. At the same time, those two documents are rendered in another context as individual documents. For this example, assume a processor generates the combined document using the generated name `combinThis-2.dita`, while the documents `combineThis.dita` and `combinedChild.dita` retain their names in their other context.
- All links in this map using the direct URI references `href="splitThis.dita"`, `href="splitThis.dita#splitThisChild"`, `href="combineThis.dita"`, or `href="combinedChild.dita"` are now ambiguous. They could go to the chunked instance from this map, or to the individual topics in the other context. Implementations will have to guess...
which topic to target: the split or combined instances from this map or versions in the alternate context from the root map.

- All links using indirect key-based references keyref="splitThisKey" or keyref="splitThisChildKey" are also ambiguous, because the key definitions are not associated explicitly with the chunked or not-chunked instance. If key scopes are used, applications might more reliably guess that the intended target is the split copy in this map, but this is not guaranteed.

- All links using keyref="explicitSplitKey", keyref="combinedThisKey", or keyref="combinedChildKey" are unambiguous; they can only resolve to the chunked instance from this submap, because they are defined directly within the chunk context.

- There is no way to unambiguously link to the child document that will result from splitting splitThis.dita. This is because it is only possible for the element using @chunk to associate a key definition with the first or root topic in the document. While other key definition elements can be used to associate keys with other topics in the same document, that can only be done outside of the navigation context that uses @chunk; as a result, a processor cannot guarantee whether the intended link target is the split topic from the @chunk context, or a use of the same topic in the second context. It is possible for an implementation to define its own way to resolve this ambiguity; however, if a situation requires both multiple instances of split topics and unambiguous cross-implementation links to those split topics, alternate reuse mechanisms need to be considered.
# A Aggregated RFC-2119 statements

This appendix contains all the normative statements from the DITA 2.0 specification. They are aggregated here for convenience in this non-normative appendix.

<table>
<thead>
<tr>
<th>Item</th>
<th>Conformance statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>001 (3)</td>
<td>The following rules apply to all values of the @chunk attribute:</td>
</tr>
<tr>
<td></td>
<td>• When the source document organization has no effect on published output, such as when producing a single PDF or EPUB, processors <strong>MAY</strong> ignore the @chunk attribute.</td>
</tr>
<tr>
<td></td>
<td>• When the @chunk attribute results in more or fewer documents based on the combine or split tokens, the hierarchy of topics within the resulting map and topic organization <strong>SHOULD</strong> match the hierarchy in the original topics and maps.</td>
</tr>
<tr>
<td></td>
<td>• When the @chunk attribute results in more or fewer documents, processors <strong>MAY</strong> create their own naming schemes for those reorganized documents.</td>
</tr>
<tr>
<td></td>
<td>• @chunk attribute values apply to DITA topic documents referenced from a map. Processors <strong>MAY</strong> apply equivalent processing to non-DITA documents.</td>
</tr>
</tbody>
</table>