# Stage two: #345 New element for defining variable text

Add a new `<keytext>` element to the base and establish new rules for determining effective text content for variable text that is defined using keys

## Champion

Kristen James Eberlein, Eberlein Consulting LLC

## Tracking information

| Event | Date | Links |
|---|---|---|
| Initial suggestion | 03 December 2020 | Minutes |
| Stage 1 proposal accepted | 28 January 2020 | Minutes<br>GitHub issue |
| Stage 2 proposal submitted to TC for early feedback | 26 October 2020 | Minutes, 27 October 2020 |
| Stage 2 proposal submitted to reviewers | 30 October 2020 | Robert Anderson, Oracle<br>Dawn Stevens, Comtech Services<br>Chris Nitchie, Individual member<br>Carsten Brennecke, SAP<br>Frank Wegmann, Software AG |
| Stage 2 proposal submitted to TC | 10 November 2020 | |
| Stage 2 proposal discussed by TC | | |
| Stage 2 proposal approved by TC | | |

## Original requirement or use case

This proposal was a consequence of the development for *Proposal #16 Improvements to alternate titles*. As this proposal advanced to stage three, the champion (Chris Nitchie, individual member) uncovered previously unnoticed technical issues. Issue #16 was returned to stage two; issue #345 was added to cover the new element for defining variable text and revised rules for determining effective text content. *Proposal #16 Improvements to alternate titles* was re-approved at stage two on 21 April 2020.

## Use cases

DITA users need to define variable text using key references. This functionality was added in the DITA 1.2 release, but the rules outlined in the specification for how such references are resolved were complex. The specification was edited thoroughly for the DITA 1.3 release, but the rules for determining the effective text content remained complex and often yielded results that DITA users did not expect.

The following are common use cases for variable text:

- Alternate text for images
- Document titles and numbers
- File names and locations

- Link text
- Operating system names and versions
- Product and feature names
- Product slogans and tag lines
- Release dates
- Terms
- Web site URLs

These common use cases for variable text require that the content model for a new element designed for variable text definition contain the following elements (and all elements specialized from them), in addition to PCDATA:

- `<cite>`
- `<data>`
- `<keyword>`
- `<ph>`
- `<q>`
- `<term>`
- `<text>`
- `<tm>`

By adding the `<keytext>` to the base and establishing new rules for determining effective text content, the DITA TC can simplify the specification, make it easier for processors to implement the specification, and ease complexity for DITA authors and information architects who author and define variable text.

## New terminology

None

## Proposed solution

The `<keytext>` element will be added to the base vocabulary. This element will be an optional child of `<topicmeta>` in `<map>`, occurring at most once.

The rules for determining effective text content, which in DITA 1.3 were defined in *2.3.4.9 Processing key references to generate text or link text*, will change to the following:

> Processors **MUST** resolve variable text that is defined using keys by using the following sequence:
>
> 1. Effective text content is taken from the `<keytext>` element.
> 2. Effective text content is taken from the `<titlealt>` element with `@title-role` set to "linking".
> 3. Effective text content is taken from the `<titlealt>` element with `@title-role` set to "navigation".
> 4. Effective text content is taken from the title of the referenced document, if available.
> 5. Effective text content is determined by the processor.

The DITA 1.3 rules for resolving the content of the `<abbreviated-form>` will be moved to the "Rendering expectations" section in the DITA 2.0 element reference topic for `<abbreviated-form>`.

**Note** This proposal intersects with proposal #16, which establishes the creation of a new `<titlealt>` element that carries the `@title-role` attribute.

## Benefits

This proposal addresses the following questions:

**Who will benefit from this feature?**
All new DITA implementations which need to define variable text using keys; all processors that resolve variable text defined using keys; editors of the DITA specification

**What is the expected benefit?**
Ease of defining variable text using keys; easing of developing processors that resolve variable text defined using keys; simplification of the DITA specification

**How many people probably will make use of this feature?**
Many

**How much of a positive impact is expected for the users who will make use of the feature?**
Significant

## Technical requirements

### Adding new elements or attributes

The `<keytext>` will have the following content model and attributes:

#### Content model

```
<define name="keytext.content">
    <zeroOrMore>
       <choice>
          <ref name="cite"/>
          <ref name="data"/>
          <ref name="keyword"/>
          <ref name="ph"/>
          <ref name="q"/>
          <ref name="term"/>
          <ref name="text"/>
          <ref name="tm"/>
       </choice>
    </zeroOrMore>
</define>
```

The content of the `<keytext>` element is translatable.

#### Attributes

Universal attributes

### Processing impact

Processors will need to revise the processing logic that is used for resolving variable text defined using keys.

Tools and systems that manage, analyze, and report on relationships between objects will need to be updated to consider the `<keytext>` element.

### Overall usability

This proposal will make it much simpler for authors to understand how key references to variable text resolve.

## Backwards compatibility

This proposal disallows the following markup pattern that is commonly used in DITA 1.2 and 1.3:

```
<keydef keys="company-name">
  <topicmeta>
```

```
        <keywords>
          <keyword translate="no">Acme Widget Company</keyword>
        </keywords>
      </topicmeta>
    </keydef>
```

This DITA 2.0 proposal calls for replacing the above clumsy markup pattern with the simpler `<keytext>` element.

## Migration plan

If implementations have defined variable text using the `<keyword>` element, they will need to migrate to using `<keytext>` or `<linktext>` (as appropriate). This can be accomplished by the following mechanisms:

- Manual updates
- Global search-and-replace across DITA maps that include key definitions
- Simple scripting

## Costs

This proposal will have an impact on the following groups:

**Maintainers of the grammar files**

The new element will need to be added to the base.

**Editors of the DITA specification**

- One new element-reference topic is required. It will be added to "Basic map elements".
- The following existing topics will need to be edited. The numeric references in the following list are to the version 1.3, errata 02 version of the DITA specification.
  - 2.2.4.5 Reconciling topic and map metadata elements
  - 2.3.4.9 Processing key references to generate text or link text
  - 2.3.4.10.10 Example: Link modification or removal
  - 2.3.4.10.2 Examples: Key definitions for variable text
  - 2.3.4.10.3 Example: Scoped key definitions for variable text
  - 3.1.1 Base DITA elements, A to Z
  - 3.3.2.2 `<keydef>`
  - 3.10.5.2.1 `<abbreviated-form>`
  - B.6 Element-by-element recommendations for translators

**Vendors of tools**

Tool vendors will need to update the following:

- Authoring environments to support authoring of variable text using the new `<keytext>` element
- Processors to ensure correct resolution of the `<keytext>` element
- Tools and systems that manage, analyze, and report on relationships between objects

**DITA community-at-large**

Authors and information architects currently using DITA might find it disruptive to shift to using a new element for defining key-based variable text.

**Producing migration instructions or tools**

The changes required by this proposal can be covered in the planned committee note: *Migrating to DITA 2.0*.

## Examples

This section contains examples of the markup for the `<keytext>` and the new processing logic for resolving variable text that is defined using keys.

### Figure 1: Simple <keytext> element

The following code sample shows how simple variable text can be defined using the `<keytext>` element:

```
<keydef keys="company-name">
  <topicmeta>
    <keytext translate="no">Acme Widget Company</keytext>
  </topicmeta>
</keydef>
```

### Figure 2: More complex <keytext> element

The following code sample shows a variable-text definition that includes highlighting elements:

```
<keydef keys="company-name">
  <topicmeta>
    <keytext translate="no">
      <i>Super</i> Widget Squared<sup>2</sup>
    </keytext>
  </topicmeta>
</keydef>
```

### Figure 3: <keytext> provides alternate text for the image

DITA implementations often reference images using keys. In those cases, the `<keytext>` element provides the alternate text for the image:

```
<keydef keys="company-logo" href="images/logo.jpg">
  <topicmeta>
    <keytext>Blue Acorn logo</keytext>
  </topicmeta>
</keydef>
```

### Figure 4: Processing logic

The following sample shows a key definition that includes several elements within the `<topicmeta>` element:

```
<keydef href="http://www.example.com" keys="company-name">
  <topicmeta>
    <keytext>Acme Tools</keytext>
    <navtitle>Acme Tools web site</navtitle>
    <linktext>Acme Tools Web Portal</linktext>
  </topicmeta>
</keydef>
```

Once processed, the effective text content of both `<ph keyref="company-name"/>` and `<xref keyref="company-name"/>` is "Acme Tools".

To set distinct text values for both the company name and the link text that is associated with the company Web site, best practices call for using two different key definitions.