DITA 2.0 proposed feature #217 Remove @domains attribute

Remove the domains attribute, and the tokens used for the domains attribute; for specialized attributes, replace the existing parenthetical syntax with a simpler token syntax..

Date and version information

Include the following information:

Date that this feature proposal was completed
June 2019

Champion of the proposal
Robert D. Anderson

Links to any previous versions of the proposal
N/A

Links to minutes where this proposal was discussed at stage 1 and moved to stage 2
May 14 2019

Reviewers for Stage 2 proposal
TBD

Links to e-mail discussion that resulted in new versions of the proposal
xxx

Link to the GitHub issue
https://github.com/oasis-tcs/dita/issues/217

Original requirement or use case

Discussed May 14 2019 based on email to the DITA TC and extended follow-up discussion. Given the limited utility of the current domains attribute design, the high level of complexity needed to use it properly, and the mostly-theoretical benefits that have never been realized in 15 years of use, we should drop `@domains` attribute requirements that place a high level of burden on DITA architects and Implementors.

Use cases

This proposal removes extraneous features in order to simplify DITA design and implementation.

- For information architects: removing `@domains` will remove steps from creating specializations or constraints, simplifying the process for each.
- For specification readers and editors: this will remove some of the most technically complex portions of the specification (about how to set up grammar file tokens for `@domains`), particularly for specialization modules that combine structural and domain tokens.
- For DITA implementations: today there are complex rules about how to use `@domains` tokens when evaluating `@conref`; those rules are technically complex, and can confuse tool users when implemented properly. Removing those rules will simplify DITA implementations and avoid those confusing results.

In addition, this proposal replaces `@domains` with a new attribute for the one domain token that is necessary for processing and generalization. This gives a clearer purpose for the attribute (simplifying the spec for readers and simplifying DITA in general). Along with moving the value to a new attribute, this proposal simplifies the syntax for attribute domain tokens, making it easier to create that value and easier for implementations to process that value.

New terminology

N/A

Proposed solution

1. Remove definitions of `@domains`
2. Remove specification sections and topics about how to properly define tokens for `@domains`
3. Remove the grammar file definition of `@domains`
4. Define a new attribute for declaring specializations of `@props` and `@base`
5. Define a simpler syntax for declaring specializations of `@props` and `@base`

Benefits

Address the following questions:

Who will benefit from this feature?

- Information architects creating specializations or constraints
- Readers of the specification
- Maintainers of the specification
- Implementors of DITA processing tools

What is the expected benefit?

Removes one of the most technically complex portions of the specification, and also simplifies the process for evaluating domain tokens with specialized attributes.

How many people probably will make use of this feature?

- All those creating or maintaining specialization and constraint modules will have fewer steps to go through.
- Editors, reviewers, and readers of the specification will no longer have to edit / review / read the extremely complex topics about domain tokens.
- Tools that strictly comply with the specification will no longer have to work with the complex rules around `@domains` tokens.
- No impact on most DITA authors who do not create or maintain grammar file modules.

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

Medium level impact to those listed above. The rules today are relatively straightforward but require following a complicated list of rules for little or no perceived benefit, so avoiding those rules will reduce that burden for all who encounter them today.

Technical requirements

Provide a detailed description of how the solution will work. Be sure to include the following details:

Adding an attribute

- Name of the attribute: `@specializations`

  Syntax for the new attribute: today's syntax will be simplified. Today's tokens use the syntax `a(props newthing)` to indicate that `@newthing` is a specialization of `@props`; similarly, `a(props newthing newerThing)` indicates that `@newerThing` is specialized from `@newthing` which is specialized from `@props`. Instead, that syntax will be simplified to use a single token without spaces. Each attribute name is separated from its ancestor by a slash: `a(props newthing)` is simplified to `@props/newthing`, and `a(props newthing newerThing)` is simplified to `@props/newthing/newerThing`

The same syntax is used for specializations of `@base`. While that attribute does not normally provide any inheritance based processing, the token must still be defined; without it, generalization processors would not have any way to recognize and generalize the specialized attribute back into `@base`. So, where a specialization of base named `@myInfo` would define a token today using the syntax `a(base myInfo)`, with this proposal the syntax would match the one above for `@props`: `@base/myInfo`

The method for integrating these tokens into the `@specializations` attribute matches the current method for adding tokens to `@domains` using the configured grammar file shell.

Note Originally I considered using an attribute name that explicitly limited this to attribute specializations, such as `@attribute-extensions` or `@special-atts`. Based on review feedback, I've gone with a more generic name `@specializations`, which will help future proof us if DITA 2.x eventually needs to provide additional specialization tokens unrelated to attributes.

- Elements that will get the new attribute: all elements that currently take `@domains` (that is, `<map>`, `<topic>`, and their specializations)
- Processing expectations that are associated with the new attribute: same as those associated with attribute specialization tokens in `@domains` with DITA 1.3
- The attribute does not contain translatable text

Removing an attribute

- Removing `@domains` from `<topic>`, `<map>`, and all specializations of those.

Processing impact

Processors can remove support for most aspects of `@domains` processing, while processing for attribute specialization tokens must be updated to account for a new attribute name and simpler syntax.

Overall usability

No impact to authors, and much simpler for those working with domain modules.

Backwards compatibility

DITA 2.0 is the first DITA release that is open to changes affecting backwards compatibility. To help highlight any impact, does this proposal involve any of the following?

Was this change previously announced in an earlier version of DITA?

No.

Removing a document type that was shipped in DITA 1.3?

No.

Removing a domain that was shipped in DITA 1.3?

No.

Removing a domain from a document type shell was shipped in DITA 1.3?

No.

Removing or renaming an element that was shipped in DITA 1.3?

No.

Removing or renaming an attribute that was shipped in DITA 1.3?

Yes. This will impact all specialization modules and configuration shells.

Removing or replacing a processing feature that was defined in DITA 1.3?

Removing `@domains` and most associated processing (the only critical aspect of processing, for attribute domains, is moved to a new attribute).

Are element or attribute groups being renamed or shuffled?

No.

Migration plan

If the answer to any question in the previous section is "yes":

Might any existing documents need to be migrated?

The `@domains` attribute is intended to be used as a default attribute value retrieved from grammar files; wherever topics or maps that have added the attribute into a DITA document, the attribute will need to be removed.

Might any existing processors or implementations need to change their expectations?

- Processors can remove support for processing that is no longer defined in the specification, such as generalization-during-conref and differing support for loose versus strict constraints.
- Processors will need to be updated to use the new syntax in the new attribute for specialized attribute tokens.

Might any existing specialization or constraint modules need to be migrated?

Yes:

- Specializations of `<topic>` or `<map>` will need to remove declarations of `@domains`, and add a declaration for the new attribute
- Declarations of `@domains` tokens can be removed from non-attribute modules
- Declarations of `@domains` tokens for `@props` and `@base` attribute specializations will need to be modified to use a new syntax, and configuration shells might need to use a new syntax to add those tokens to the new attribute

Costs

Outline the impact (time and effort) of the feature on the following groups.

Maintainers of the grammar files

Minor impact to remove the old attribute declaration and create the new attribute + update to use the new syntax

Editors of the DITA specification

- How many new topics will be required? *None*
- How many existing topics will need to be edited? *Several, mostly by removing content*
- Will the feature require substantial changes to the information architecture of the DITA specification? *No*

Vendors of tools

Minor impact (removing extraneous processing and simplifying parsing rules for attribute domain tokens)

DITA community-at-large

- Will this feature add to the perception that DITA is becoming too complex? *No, should have the opposite effect*
- Will it be simple for end users to understand? *Yes*
- If the feature breaks backwards compatibility, how many documents are likely to be affected, and what is the cost of migration? *Few documents should be affected, most specialization modules and configuration shells will need an update.*

Producing migration instructions or tools

- How extensive will migration instructions be, if it is integrated into an overall 1.3 # 2.0 migration publication or white paper? *Minor addition to existing migration instructions*
- Will this require an independent white paper or other publication to provide migration details? *No*
- Do migration tools need to be created before this change can be made? If so, how complex will those tools be to create and to use? *Migration tools for this cannot cover all cases, because specializations /*

shells do not have to use the same format. Tools may catch some cases but some minor updates will likely be required.

Examples

With DITA 1.3, a specialization of `@props` must declare a token beginning with the letter `a` followed by a `(props` followed by a space followed by the name few the new attribute (followed by attribute names for further specializations, if present) followed by `)`. For example, if `@props` is specialized to `@newthing` which is specialized to `@newerThing` which is specialized to `@finalThing`, the DITA 1.3 domain token is `a(props newthing newerThing finalThing)`

With DITA 2.0 that syntax is simplified, removing the `a` and parenthetical grouping, as well as all spaces. The new declaration will be a single token without spaces, including the full ancestry of the specialized element – starting with `@props` and ending with the attribute name. For example, if `@props` is specialized to `@newthing` which is specialized to `@newerThing` which is specialized to `@finalThing`, the updated domain token will be `@props/newthing/newerThing/finalThing`

When a grammar file is parsed, a configured shell that defines three attribute extensions – `@deliveryTarget`, a `@props` specialization named `@newThing`, and a `@base` specialization named `@myInfo` – would include the following three tokens (not necessarily in this order):

```
specializations="@props/deliveryTarget @props/newThing @base/myInfo"
```