Using Markdown and Lightweight DITA in a Collaborative Environment

Leigh White and Keith Schengili-Roberts - April 24, 2017

A Bit About Us

Leigh W. White

- Background in Theoretical Linguistics
- In Tech Comm since 1996
- Working with XML since ~ 2001
- Working with DITA since ~ 2007
- At IXIASOFT since 2013
- Author of *DITA For Print: A DITA Open Toolkit Workbook*
- Contributor to *The Language of Technical Communication* and *The Language of Content Strategy*

A Bit About Us

Keith Schengili-Roberts

- Working in Tech Comm since early 1990s
- Working with DITA since 2004 (pre-DITA 1.0)
- Was IXIASOFT's first customer (AMD); working with the firm since 2015
- Chair of OASIS DITA Adoption Committee, member of LwDITA SC and DITA TC
- Author of four technical titles, contributor to forthcoming *Current Practices and Trends in Technical Communication*

What We'll Cover

- Lightweight DITA (LwDITA) background and development
- Markdown as entry to LwDITA
- One typical Markdown to LwDITA workflow using the IXIASOFT DITA CMS
  - There are many!
  - Import/export and round-tripping concerns
- Demo

Let's get started!

What is LwDITA?

- It is “a slimmed-down version of the Darwin Information Typing Architecture. It is designed to ease adoption and implementation of DITA.”
- Currently three different versions:
  - XDITA (LwDITA using XML)
  - HDITA (LwDITA using HTML5)
  - MDITA (LwDITA using Markdown)
- There is no official specification (yet), but there is considerable info available, enough to start working with it

The Evolution of LwDITA

- First publicly mentioned by Michael Priestley (a co-founder a DITA) in a presentation at DITA Europe 2013
- LwDITA goals:
  - Reduce complexity by stripping down DITA tagset to its essentials
  - Design would allow SMEs to more easily create content
  - Ease adoption of DITA learning curve by new groups
- This is the birth of XDITA

LwDITA and HTML5

- Subsequent blog post (2014) on dita.xml.org by Michael Priestley that first outlined authoring LwDITA based on HTML5 principles = **HDITA**
- Outlined parallel structure for XDITA and HDITA
  - This sparked a further idea… does DITA need to be bound to XML?

dita xml·org

Online community for the Darwin Information Typing Architecture OASIS Standard

HOME | WIKI KNOWLEDGEBASE | NEWS | EVENTS | PRODUCTS | SERVICES | RESOURCES | FO

Overview of Lightweight DITA (XDITA and HDITA)

View | Revisions

Blog entry: Submitted by Michael Priestley on Fri, 2014-04-11 16:36. Last updated on Tue, 2014-05-06 14:38.

The goal of this proposal is to align a lightweight DITA profile in XML with an equivalent markup specification based on HTML5. This is not a complete specification, just something to start the discussion going. There's still lots of room for change, as well as for adding specific mappings for additional semantics for learning and training content, epubs, or other formats.

While XML-based publishing chains remain the industry standard for many content-centric industries (such as publishing, pharmaceutical, and aerospace), concerns have been raised about their complexity, especially as a barrier to new adopters or contributing authors.

HDITA	XDITA
<topic>	<article>
<title>	<h1> (in <article>) or <h2> (in <section>)
<shortdesc>	<p>
<body>	No equivalent (ignored)
<section>	<section>
<dl>	<dl>

Tagless DITA?

- Carlos Evia first suggested using Markdown in LwDITA
- Jarno Elovirta devised DITA-OT plugin for Markdown
- This idea was presented fully at DITA NA 2015, with introduction of the idea of Markdown-based DITA = MDITA

So Where is LwDITA Today?

- Being driven by OASIS Lightweight DITA Sub-committee
- Two draft Committee Notes are currently in development:
  - **Authoring Elements:** Outlines rationale for LwDITA, describes how XDITA, HDITA and MDITA "tags" are intended to work
  - **Template-based Specialization:** Separate document outlining how a simplified, template-based specialization mechanism for LwDITA ought to work
- Committee Notes != Specification
  - …but they will outline the path to be taken

LwDITA vs. “Full” DITA

DITA 1.3 All-inclusive:

- 26 document types, 621 elements

DITA 1.3 Base:

- 4 document types, 189 elements

Lightweight DITA:

- 1 document type, 46 elements

Some General LwDITA Guidance

- LwDITA content is still valid DITA and can be incorporated into “full” DITA
- No automatic “round-tripping” between DITA and LwDITA
- Just map, no bookmap
- Mixed content not allowed; all text must be in a `<p>`
- No CALS table elements (i.e. `<table>`, `<row>`, `<entry>`, etc.)
- It is really “DI~~T~~A”, as content is not typed (result is a generic topic)

LwDITA Bonus: Multimedia Controls!

- Current draft of the Committee Note includes eight additional elements, most of which are focused on multimedia
- None of these elements currently exist in DITA 1.3 (but may in DITA 2.0)
- Ability to add multimedia sound/video content in line with HTML5

Multimedia:

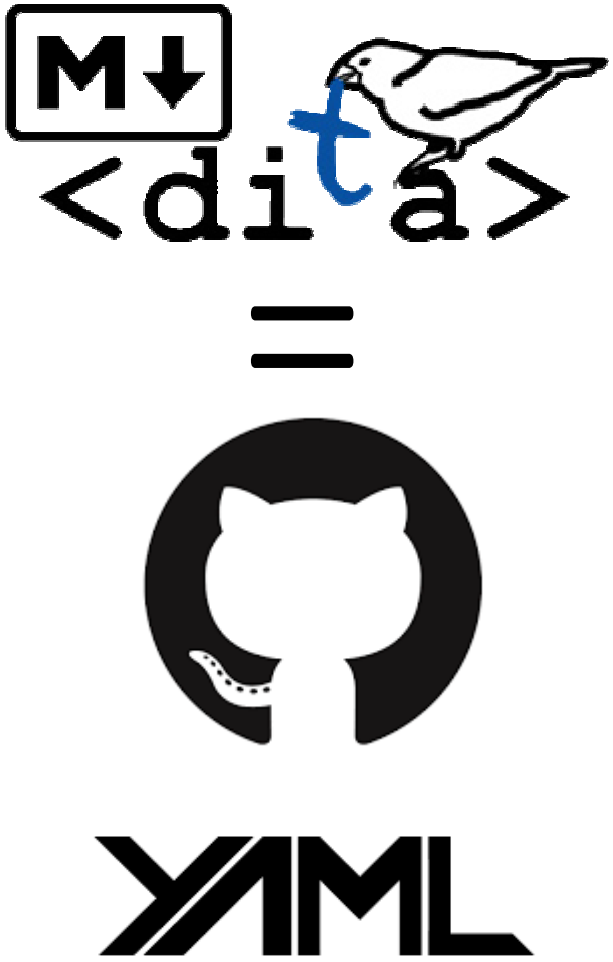
<audio>
<controls>
<fallback>
<poster>
<source>
<track>
<video>

Also:

<fnref>

How MDITA Aligns with Markdown

- There is no single standard for Markdown
- Base MDITA is derived from GitHub-flavored Markdown (GFM) with tables as an option
- "Valid, extended" MDITA includes GFM, YAML headers, and HDITA elements where no equivalent exists in Markdown
  - e.g. `<note>` becomes `<p data-hd-class="note">`



Simple GHF MDITA Topic Example

capabilities_and_advantages.md - MarkdownPad 2

File Edit Insert View Tools Help

capabilities_and_advantages.md

YAML Header

```
---
id: capabilities_and_advantages
shortdesc: A brief overview of the additional features that adding the expansion interface to your TRS-80.
author: Reginald Sutton
---

# Capabilities and Advantages

The Interface allows you to add the following Radio Shack modules to your system:

1. Screen Printer (26-1151)
2. Line Printer (26-1150)
3. Mini-Disk System (26-1160/26-1161)
4. Cassette Recorder number 2 (14-841)

The Screen Printer and Line Printer allow you to obtain hard copy (printed) information generated by your TRS-80.

The TRS-80 Mini-Disk System is a small version of the floppy disk. It provides vast storage space and much quicker
access time than tape. The number 1 disk contains about 80,000 bytes of free space for files. Each additional disk
has 89,600 bytes of file space. The Disk System has its own set of commands that allow manipulation of files and
expanded abilities in file use. The TRS-80 Mini-Disk System uses sequential or random access. The disks will allow
use of several additional LEVEL II commands.

<p data-hd-class="note">Because of the presence of a Disk Controller in the Expansion Interface, the computer will
try to input the additional commands.</p>

When the Expansion Interface is connected to the computer, it assumes that a Mini-Disk is connected. To use the
Expansion Interface without a Mini-Disk, press the BREAK key on the TRS-80 keyboard. This will override the Mini-
Disk mode and allow normal LEVEL II operation.

The use of two cassettes allows a much more efficient and convenient manner of updating data stored on tape. For
example, if you have payroll data stored on tape, the information can be read, one item at a time, from Cassette
Recorder number 1, then changed or added to and written out on Cassette Recorder number 2. The example cited is a
very simple application; however, very powerful routines can be constructed to allow input and output of data using
two tapes simultaneously.

<p data-hd-class="note">This unit is designed to be used with Level II only. Do not use with level I.</p>

![Image](figure_1.jpg)

FIGURE 1. Expansion Interface.*

| * Catalog Number | Description | RAM |
|---|---|---|
| 26-1140 | TRS-80 Expansion Interface | 0K |
| 26-1141 | TRS-80 Expansion Interface | 16K |
| 26-1142 | TRS-80 Expansion Interface | 32K |
```

Title

Numbered List

Image

Table

Words: 372 Characters: 2382

Capabilities and Advantages

The Interface allows you to add the following Radio Shack modules to your system:

1. Screen Printer (26-1151)
2. Line Printer (26-1150)
3. Mini-Disk System (26-1160/26-1161)
4. Cassette Recorder number 2 (14-841)

The Screen Printer and Line Printer allow you to obtain hard copy (printed) information generated by your TRS-80.

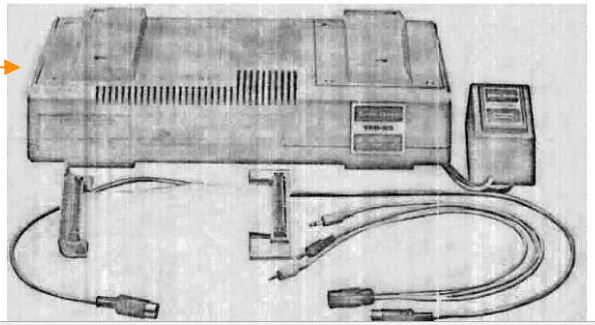
The TRS-80 Mini-Disk System is a small version of the floppy disk. It provides vast storage space and much quicker access time than tape. The number 1 disk contains about 80,000 bytes of free space for files. Each additional disk has 89,600 bytes of file space. The Disk System has its own set of commands that allow manipulation of files and expanded abilities in file use. The TRS-80 Mini-Disk System uses sequential or random access. The disks will allow use of several additional LEVEL II commands.

Because of the presence of a Disk Controller in the Expansion Interface, the computer will try to input the additional commands.

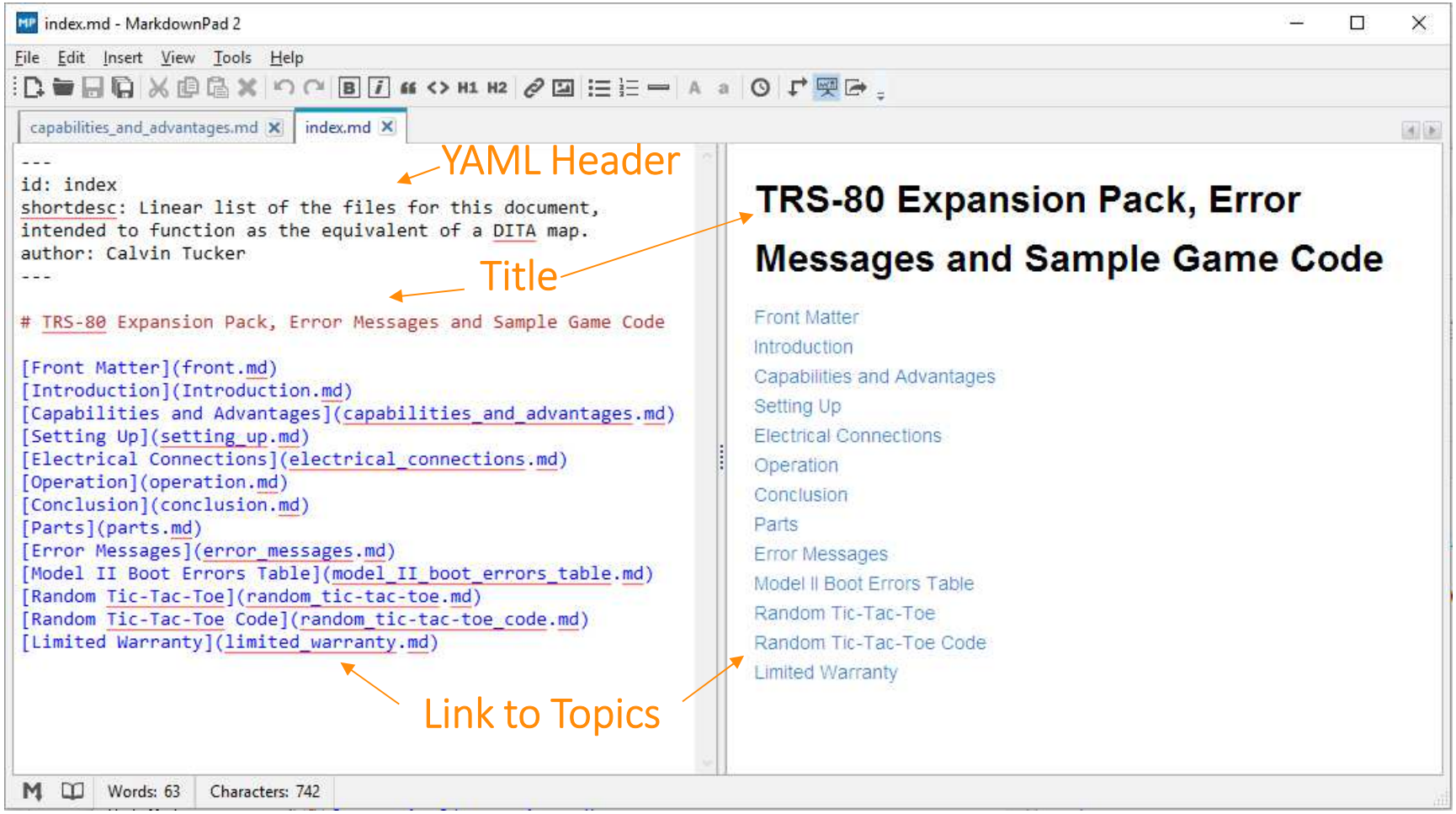
When the Expansion Interface is connected to the computer, it assumes that a Mini-Disk is connected. To use the Expansion Interface without a Mini-Disk, press the BREAK key on the TRS-80 keyboard. This will override the Mini-Disk mode and allow normal LEVEL II operation.

The use of two cassettes allows a much more efficient and convenient manner of updating data stored on tape. For example, if you have payroll data stored on tape, the information can be read, one item at a time, from Cassette Recorder number 1, then changed or added to and written out on Cassette Recorder number 2. The example cited is a very simple application; however, very powerful routines can be constructed to allow input and output of data using two tapes simultaneously.

This unit is designed to be used with Level II only. Do not use with level I.



Simple GHF Map Example



Interest in LwDITA from the DITA Listening Sessions

LwDITA came up many times during the OASIS DITA Listening Sessions (hosted by DITA Adoption):

- Most were interested in using it to gather information from SMEs, typically software developers
- At least two firms were already using it in production

One Typical Workflow [1]

1. SMEs or developers/engineers author Markdown in a third-party system
   - Could be procedures, troubleshooting, extracted code samples, error messages, etc.
   - Type of information could drive import process and organization within DITA CMS... decisions to be made!
2. Markdown is converted to LwDITA* outside of CMS
   - Currently no way to dynamically convert upon import, but not out of the question for future
   - Could happen in various ways... we will show use of Markdown and Normalize DITA-OT plugins

*Does not have to be LwDITA per se... we'll get to that

One Typical Workflow [2]

3. Converted DITA is imported into the DITA CMS
   - Can be imported as read-only
   - Can be imported to overwrite previously-imported versions of the same content or to create new content
   - Doctypes of imported content can be retained as-is or refactored as needed

One Typical Workflow [3]

4. Edit imported content as necessary (if not read-only)
5. Add topics to maps as appropriate
6. Reference from topics as conrefs, conkeyrefs, keys, cross-refs, etc. as appropriate
7. Send content through review/approval cycle as appropriate
8. Output deliverables
9. Rinse and repeat

Markdown to LwDITA: A Must?

- Converting Markdown to LwDITA is a safeguard against introducing elements that would not round-trip back to Markdown
- If roundtripping is not a concern, convert to full DITA
  - Possibly a distinction without a difference since Markdown markup is limited and therefore will be converted to limited DITA tagset anyway
- You determine whether content is imported into CMS as full DITA or LwDITA

Third-party System Export Concerns

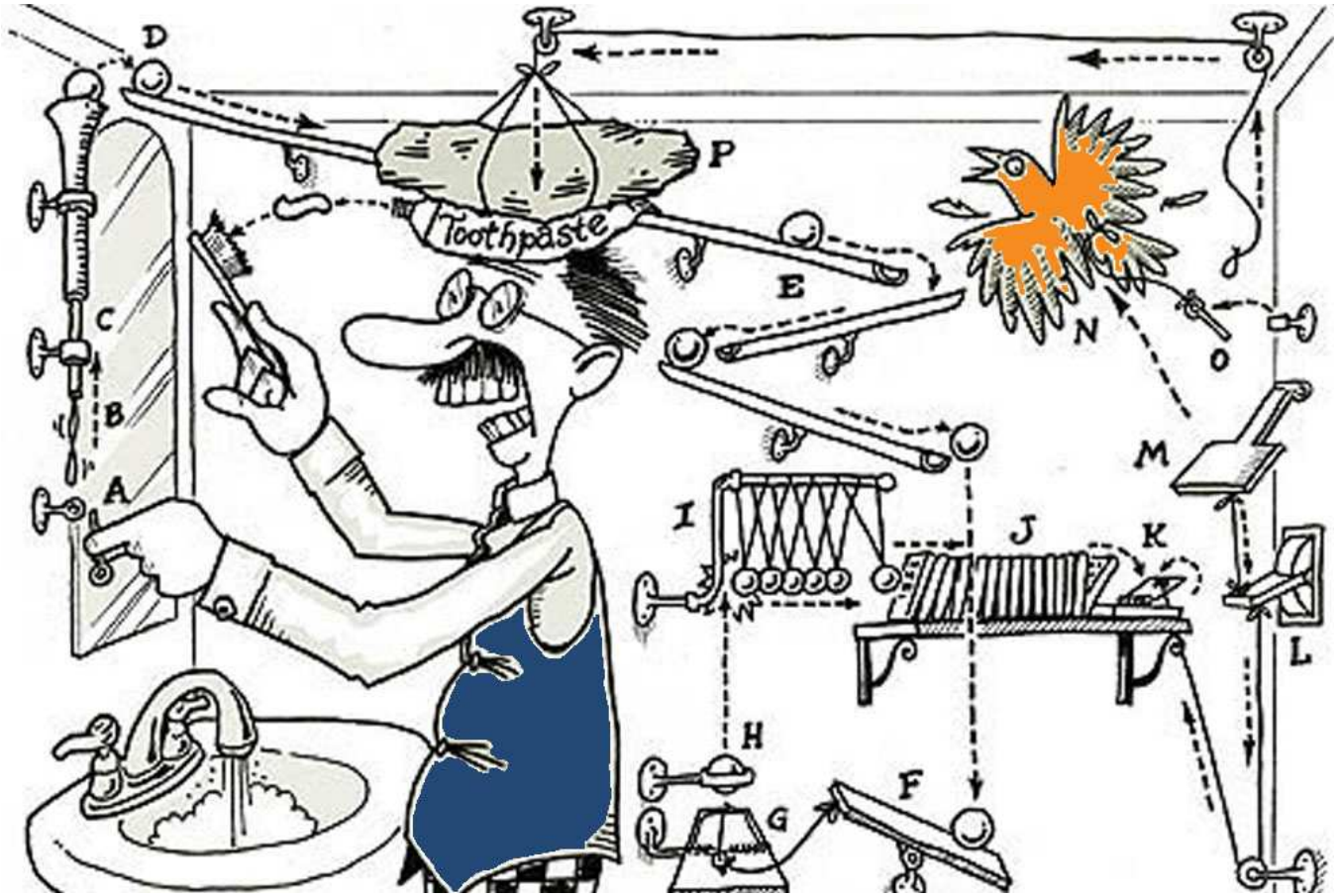
Can external conversion process guarantee static file names and IDs from cycle to cycle?

- If file name is dynamic and different from export to export, the CMS cannot match incoming content to existing content to overwrite...will create new content
  - Existing references will refer to old content, not new content
- If internal IDs are dynamic and different from export to export, the CMS will overwrite previous IDs with new ones
  - Existing references will be broken
  - Import is a simple overwrite, not an intelligent compare/merge

Round-tripping: DITA CMS Concerns

- Upon import, DITA CMS refactors file names to *aaannnnnnnnnnnn.xml* format
  - DITA CMS retains original file name in object properties
    - This is used for content matching in subsequent imports of same content (and optionally, original file path)
  - Content is exported from DITA CMS with refactored file name
    - Export *could* be configured to revert to original file names but not currently part of DITA CMS
- What kind of matching/overwriting can third-party system do on import?

DEMO



MDITA a Solution for Easing Content Conversion?

- One other possible scenario for MDITA -> DITA use is converting text-based content where no other source file exists
  - e.g. have a PDF of a document but no longer have the source file that produced it
- Can take text from the file, add YAML headers, plus search-and-replace routines to add MDITA content. Then use a textfile-splitting program to process the result into individual MDITA files.
  - These files can then be imported into “full” DITA where additional processing (i.e. content typing, use of full range of DITA elements) can be done

In summary / Questions

- LwDITA not finalized but far enough along for use
- Collaborative workflow should be one-way, don’t attempt roundtripping
- Content modeling of imported content is a must
- File names and id's of content exported from third-party system must remain static
- DITA CMS not concerned with conversion method; only that resulting content is valid DITA
- DITA CMS does not include dynamic conversion of content upon import
- Plugins for converting Markdown to DITA are currently available

QA

Blog: www.ixiasoft.com/en/news-and-events/blog

Twitter: @IXIASOFT (plus @KeithIXIASOFT and @LeighWW)

IXIASOFT DITA CMS Users LinkedIn group: www.linkedin.com/groups?gid=3820030