



Creating Unified Content Experiences

October 27-30, 2019 Portland, Oregon

When DITA meets Markdown

@AlexJitianu

Agenda

- Content and markup
- Structured authoring
- Markdown
- DITA
- Markdown meets DITA in a docs project
- Docs as Code, CI/CD

Tools Warning



It all starts with the content

Create a Google account

How to create or set up your Google Account on your mobile phone.

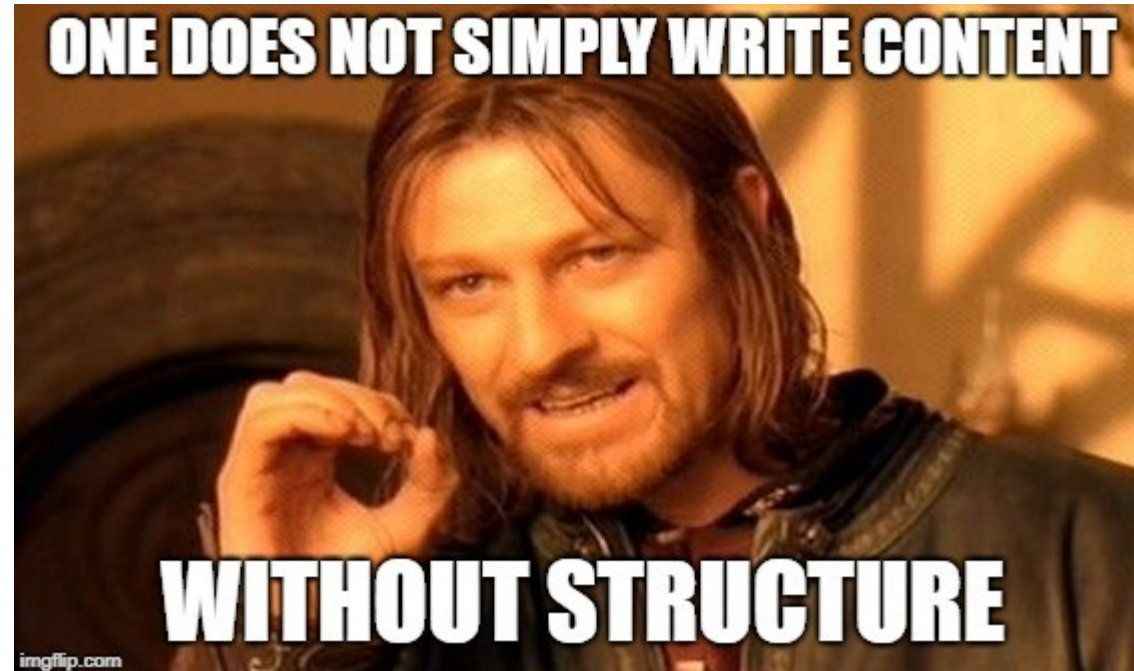
From a Home screen, swipe up to access Apps.

Tap Settings > Accounts

Tap Add account > Google.

A typical task

Content alone is not enough



Why do we need structure?

- Defines the organization/model of content
- Helps us enforce the defined model
- Increases consistency
- Automatic processing
- Faster publishing workflows

What Meaning Lies Beneath

Create a Google account

Title

How to create or set up your Google Account on your mobile phone.

Short description

From a Home screen, swipe up to access Apps.

Tap Settings > Accounts

Tap Add account > Google.

Procedure

Encode it with a Markup Language

Create a Google account

Title

How to create or set up your Google Account on your mobile phone.

Short description

From a Home screen, swipe up to access Apps.

Tap Settings > Accounts

Tap Add account > Google.

Procedure

Markdown

- Easy to learn
- Minimalistic
- Many authoring tools available
- Publishing tools

Create a Google account

=====

How to create or set up your **Google Account** on your mobile phone.

* From a Home screen, swipe up to access Apps.

* Tap **Settings** > **Accounts**

* Tap **Add account** > **Google**.

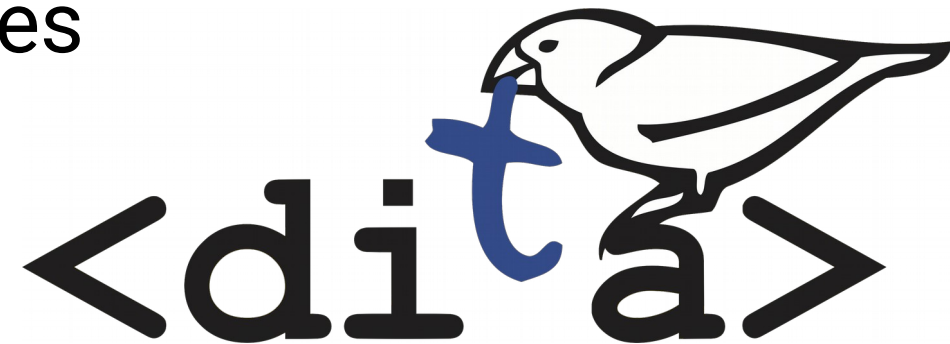


XML

- e(**X**)tensible (**M**)arkup (**L**)anguage
- Define your own tags/markup
- Powerful mechanisms to enforce valid content structure (DTD/Schema/Schematron)

DITA

- DITA is an XML-based open **standard** for structuring, developing, managing, and publishing content.
- Semantic markup (separates formatting from content)
- Strong content reuse concepts
- Restrictions and specializations
- Huge ecosystem of publishing choices



Side by side

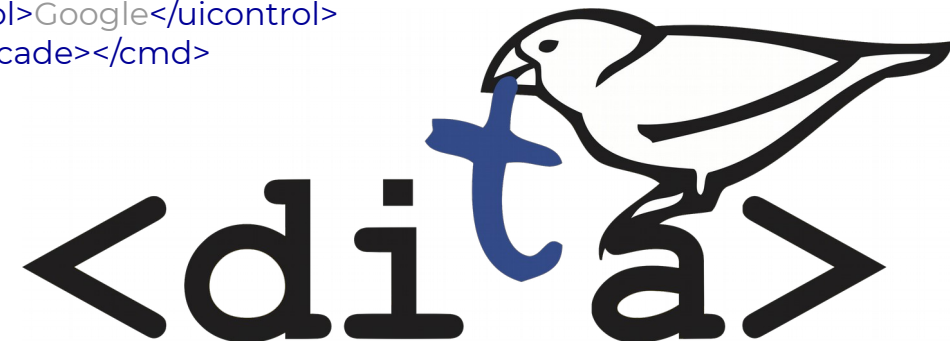
Create a Google account

=====

How to create or set up your **Google Account** on your mobile phone.

1. From a Home screen, swipe up to access Apps.
1. Tap **Settings** > **Accounts**
1. Tap **Add account** > **Google**.

```
<task id="create_google_account">
  <title>Create a Google account</title>
  <shortdesc>How to create or set up your <term>Google Account</term>
on your mobile phone.</shortdesc>
  <taskbody>
    <steps>
      <step>
        <cmd>From a Home screen, swipe up to access Apps.</cmd>
      </step>
      <step>
        <cmd>Tap <menucascade>
          <uicontrol>Settings</uicontrol>
          <uicontrol>Accounts</uicontrol>
        </menucascade></cmd>
      </step>
      <step>
        <cmd>Tap <menucascade>
          <uicontrol>Add account</uicontrol>
          <uicontrol>Google</uicontrol>
        </menucascade></cmd>
      </step>
    </steps>
  </taskbody>
</task>
```



Side by side – visual mode

Create a Google account

How to create or set up your **Google Account** on your mobile phone.

1. From a Home screen, swipe up to access Apps.
2. Tap **Settings** > **Accounts**
3. Tap **Add account** > **Google**.

Create a Google account

Short Description: How to create or set up your *Google Account* on your mobile phone.

1. From a Home screen, swipe up to access Apps.
2. Tap **Settings**→**Accounts**
3. Tap **Add account**→**Google**



Scenario

- Main documentation project written in DITA
- Contributors (devs) sending content in Markdown

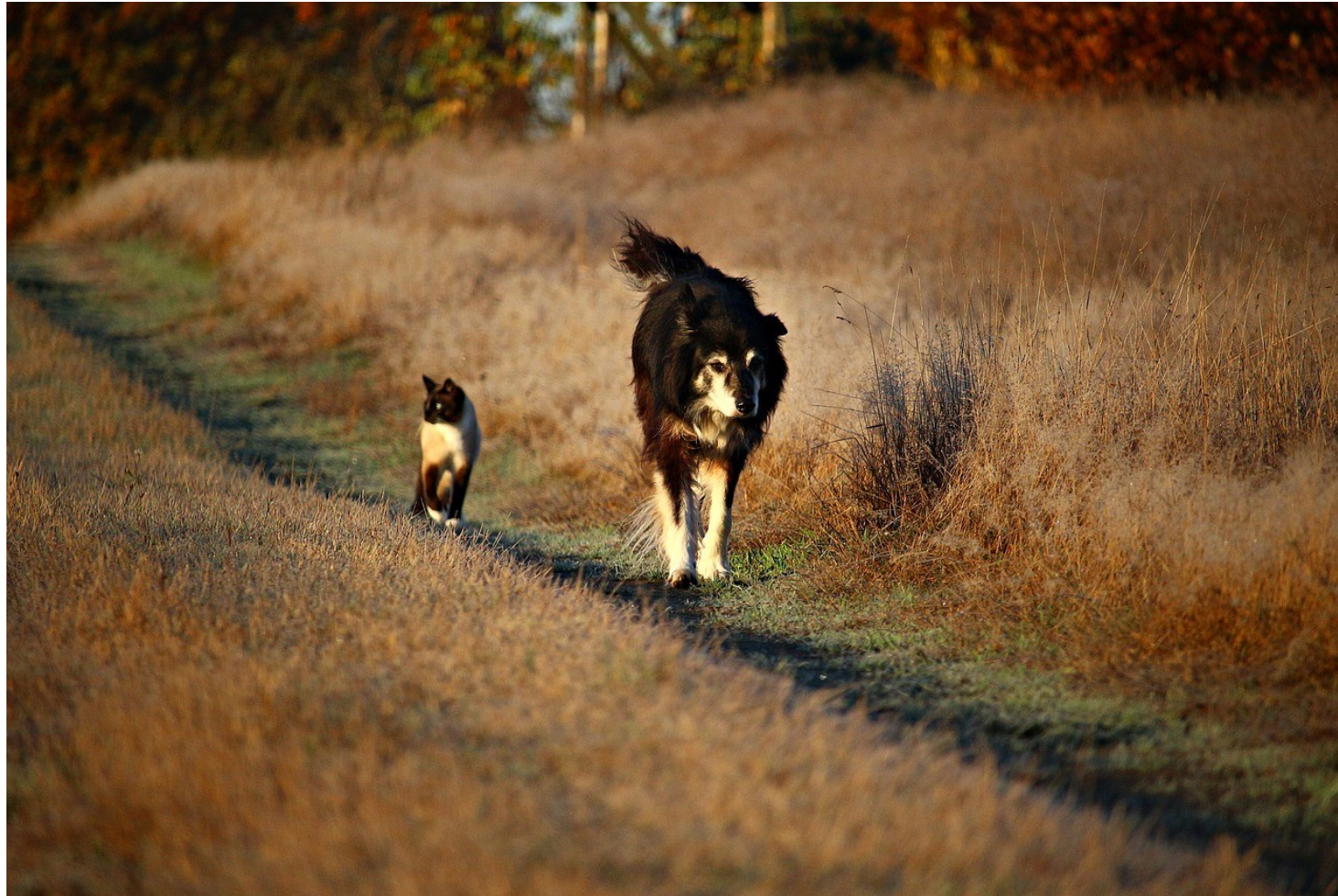
Will they be friends?



Will they be foes?



The journey begins...



[1] Convert MD to DITA

- MD => HTML => DITA
 - <https://pandoc.org/>
 - <http://dita-ot.sourceforge.net/1.5.2/readme/DITA-h2d-ant.html#h2d-ant>
- MD => DITA
 - DITA-OT plugin developed by Jarno Elovirta
<https://github.com/jelovirt/dita-ot-markdown>
 - DITA to DITA plugin
<https://github.com/dita-ot/org.dita.normalize>
 - Oxygen Batch Converter plugin
<https://github.com/oxygenxml/oxygen-resources-converter>

Hands-on time

– Prerequisites:

- Java SE Runtime Environment 8
 - <https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>
- DITA-OT
 - <https://github.com/dita-ot/dita-ot/releases/download/3.3.4/dita-ot-3.3.4.zip>
- DITA to DITA plugin
 - <https://github.com/dita-ot/org.dita.normalize>
- Markdown sample file



```
cd dita-and-markdown
```

```
dita-ot-3.3.4\bin\dita.bat -i demo-files\conversion\sample.ditamap -format dita -o out
```

[2] Keep MD and use it in DITA

- Dynamic conversion (custom URL)

- <https://en.wikipedia.org/wiki/URL>

Scheme

Domain name

Path

- `<topicref href="md2dita:/topic.md" format="dita"/>`

- <https://github.com/oxygenxml/dita-glass>

[2] Keep MD and use it in DITA

- Dynamic conversion (custom URL)
 - `<topicref href="md2dita:/topic.md" format="dita"/>`
 - <https://github.com/oxygenxml/dita-glass>
- Refer MD files directly in your map
 - `<topicref href="tasks/changingtheoil.md" format="markdown"/>`
- Seamless publishing

Hands-on time

- Using specific DITA concepts in MD
 - Metadata
 - Specialization types
 - Titles and document structure
 - Image and Key references



Syntax reference: <https://github.com/jelovirt/dita-ot-markdown/wiki/Syntax-reference>

Work on: dita-and-markdown\demo-files\hybrid-project\markdown-dita\tasks\washingthecar.md

Hands-on time

YAML Header

Washing the car {#changing.the.oil} **Pandoc Attributes**

Keep your car looking great by washing it regularly.

1. Move the car onto the driveway. Avoid washing the car in d
1. Attach the water hose to a spout and pull the free end ove
1. Fill a bucket with soapy water.
1. Use a sponge to apply the soapy water to the car and gentl
1. Rinse the car by spraying clean water from the hose.
1. Dry the car using a dampened chamois.

An image reference that uses the href attribute.

Image Reference

See also Changing the oil and waterhose topics

Cross reference and Key reference



What is Lightweight DITA?

- LwDITA is a proposed standard for expressing *simplified DITA documents* in *XML, HTML5, and Markdown*.
- The core goals of LwDITA:
 - Provide a simpler DITA experience
 - Provide mappings between *XML, HTML5, and Markdown* that enable individuals to:
 - Author content in the format of their choice
 - Easily exchange and publish content whose source exists in these different markup languages

What is Lightweight DITA?

- LwDITA is a proposed standard for expressing *simplified DITA documents* in *XML, HTML5, and Markdown*.
- The core goals of LwDITA:
 - Provide a simpler DITA experience
 - Provide mappings between *XML, HTML5, and Markdown* that enable individuals to:
 - Author content in the format of their choice
 - Easily exchange and publish content whose source exists in these different markup languages

[2] Keep MD and use it in DITA

- Dynamic conversion (custom URL)
 - `<topicref href="md2dita:/topic.md" format="dita"/>`
 - <https://github.com/oxygenxml/dita-glass>
- Refer MD files directly in your map
 - `<topicref href="tasks/changingtheoil.md" format="markdown"/>`
 - `<topicref href="tasks/changingtheoil.md" format="mdita"/>`
- Seamless publishing

Hands-on time

- Working on LWDITA topics

- YAML header
- Short descriptions
- Reuse
 - Content references - @data-conref
 - Key reference - [keyref]
 - Linking
 - Variable text - [keyref]

```
---  
id: install-and-setup  
author: Kevin Lewis  
---
```



Work on: dita-and-markdown\demo-files\hybrid-project\lwdita-project\mdita-topics\basic-concepts.md

Hands-on time



YAML Header

```
# Basic Concepts of Network Lighting
```

Variable

```
Network light bulbs from your "Remote Network Lighting" work with your light fixtures the same way
```

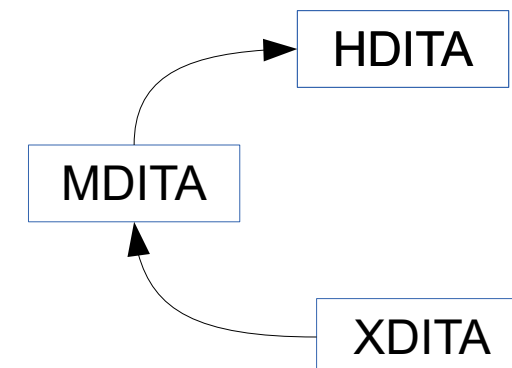
- The lighting element in the light bulb uses energy-efficient LED technology.
- The light bulb includes wireless technology that allows the light bulb to connect to a network.

```
Make sure power to the fixture where you are installing the light bulb is turned
```

Target for content reference

```
Even in low power networks, be sure to disconnect all devices before performing maintenance tasks.
```

Content reference



Advantages

- Single sourcing across DITA and Markdown
- Collaboration on Markdown source
- Use DITA features and publishing options with Markdown
- Use Markdown publishing options
 - <https://github.com/jelovirt/dita-ot-markdown#generating-markdown-output>
 - Make use of publishing platforms like Jekyll, Vuepress, Mkdocs etc.
 - <https://nostalgic-hamilton-b6dae0.netlify.com/>

Disadvantages

- Markdown lacks semantics
 - <https://github.com/IBM-Cloud/docs-services/tree/staging#using-the-copyright-and-last-updated-header-required>
 - https://raw.githubusercontent.com/IBM-Cloud/docs-services/staging/getting_started_template/servicename_task.md

Disadvantages

- Markdown lacks semantics
- Consistency challenges (not a standard, can receive different flavors)
 - <https://github.com/IBM-Cloud/docs-services/tree/staging#using-the-copyright-and-last-updated-header-required>
 - https://raw.githubusercontent.com/IBM-Cloud/docs-services/staging/getting_started_template/servicename_task.md

Disadvantages

- Markdown lacks semantics
- Consistency challenges (not a standard, can receive different flavors)
- Markdown language restrictions
- No reuse mechanisms (unless you go with LwDITA)

Disadvantages

- Review/Collaboration tracking challenges
 - *When done on the Markdown source*
 - No support for tracking changes
 - Difficult to visualize changes (diffs needed)
 - *When done on converted content, like PDF*
 - Writer Generate PDF => Devs review on PDF => Writer incorporates Review
 - Extra overhead to incorporate review into source

Markdown Consistency Challenges

- What can we do?

Vale

- A syntax-aware linter for prose built with speed and extensibility in mind. <https://errata-ai.github.io/vale/>
- Supports plain text, markup (**Markdown**, reStructuredText, AsciiDoc, and HTML)
- YAML-based extension system



Vale

- Support the following types of rules/styles:

- Existence
 - extends: existence
 - message: "Don't use end punctuation in headings."
- Substitution
 - link:
<https://docs.microsoft.com/en-us/style-guide/punctuation/periods>
- Occurrence
 - nonword: true
- Repetition
 - level: warning
 - scope: heading
-
 - tokens:
 - '[a-z0-9][?!](?:\s|\$)'

<https://errata-ai.github.io/vale/styles/#creating-a-style>

<https://github.com/errata-ai/Microsoft/blob/master/Microsoft/HeadingPunctuation.yml>

Hands-on time

- Let's create and run a Vale rule
 - *"Use the Oxford comma in a list of three or more items."*



Markdown Consistency Challenges

- XML/DITA has Schematron
- It does structure checks too

```
<sch:pattern>  
  <sch:rule context="topic">  
    <sch:assert test="shortdesc">Please add a short description.</sch:assert>  
  </sch:rule>  
</sch:pattern>
```



Hands-on time

- Let's create and run a Schematron rule
 - *"A short description is mandatory."*



Schematron for Markdown

- Markdown syntax maps to a subset of HTML tags
- Apply Schematron on the HTML with back-mapping support

Why do Devs tend to use Markdown?

- It has a low learning curve
- You do things fast
- Don't have time to learn another language
- They don't need any additional tool installed on their system

[3] Could Devs write DITA?

- Learn it as you use it through Markdown2DITA controlled conversions. Like a *“Learning assistant”*.
 - Powered by Schematron Quick Fixes
 - <https://github.com/oxygenxml/ditaMark>

[3] Could Devs write DITA?

- Learn it as you use it through Markdown2DITA controlled conversions. Like a *“Learning assistant”*.
 - Powered by Schematron Quick Fixes
 - <https://github.com/oxygenxml/ditaMark>
- Give Devs specialized editing environments (cloud based):
 - Specialized UI (e.g. HTML form) that generates consistent DITA
 - Specialized Web based DITA editors
 - *Oxygen XML Web Author*
 - *Xeditor*
 - *Fonto*

Could Devs write DITA?

Task template

Title

Create a Google account

Short Description

How to create or set up your Google Account on your mobile phone.

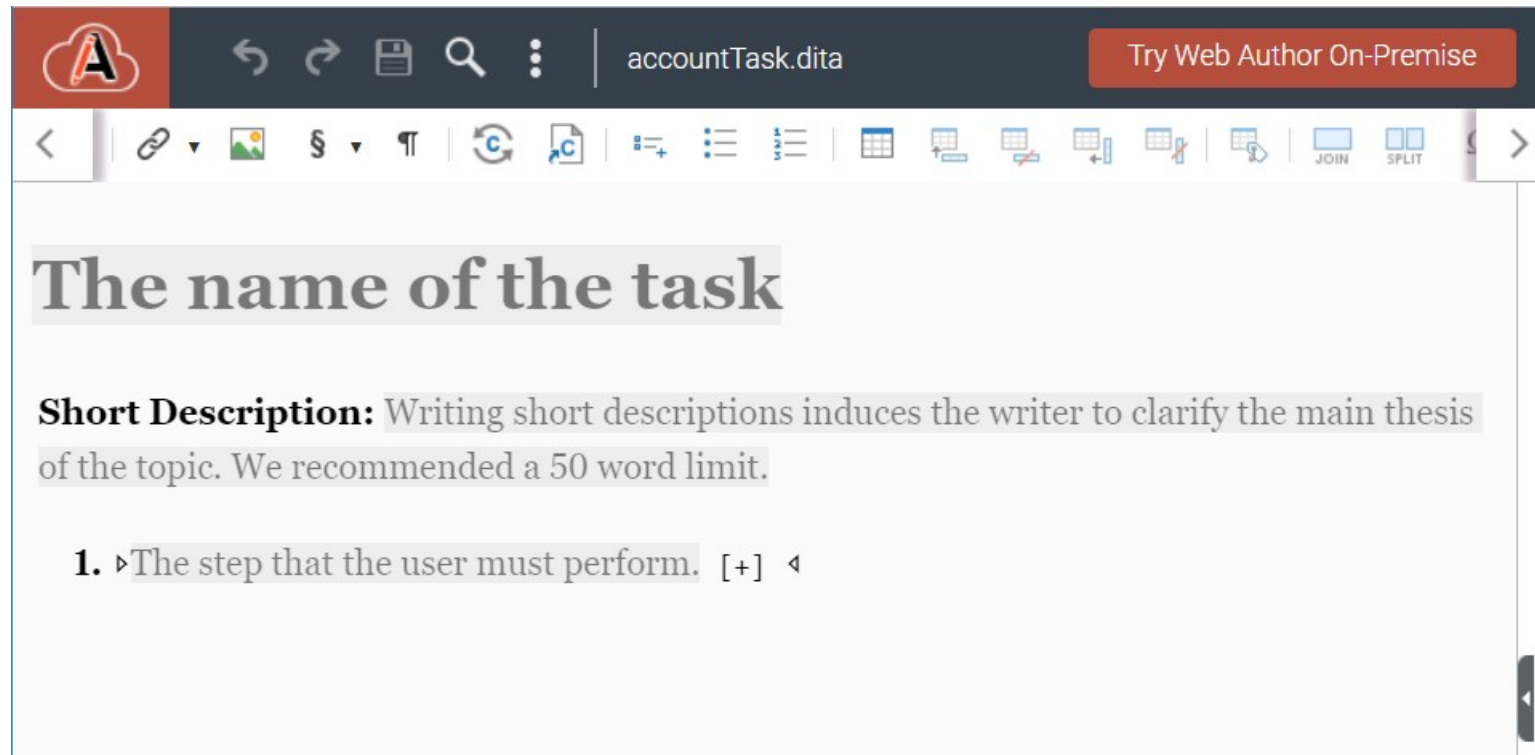
Step

From a Home screen, swipe up to access Apps.

+

Submit

Could Devs write DITA?



Which way to go?

- There is no one-size-fits-all solution
- Convert if it's a one time thing
- Keep them together, achieve single sourcing
- Consistency/Collaboration might require a switch to DITA



Docs as Code

- Refers to a philosophy that you should be writing documentation with the same **tools** and **workflows** as code:
 - Version Control
 - Collaboration and Review Process
 - Automated Tests, Builds, Delivery
 - Issue Trackers
 - Plain Text Markup

Choosing a Suitable Text Markup

- **Markdown**
- Asciidoc
- XML
 - Docbook
 - **DITA**

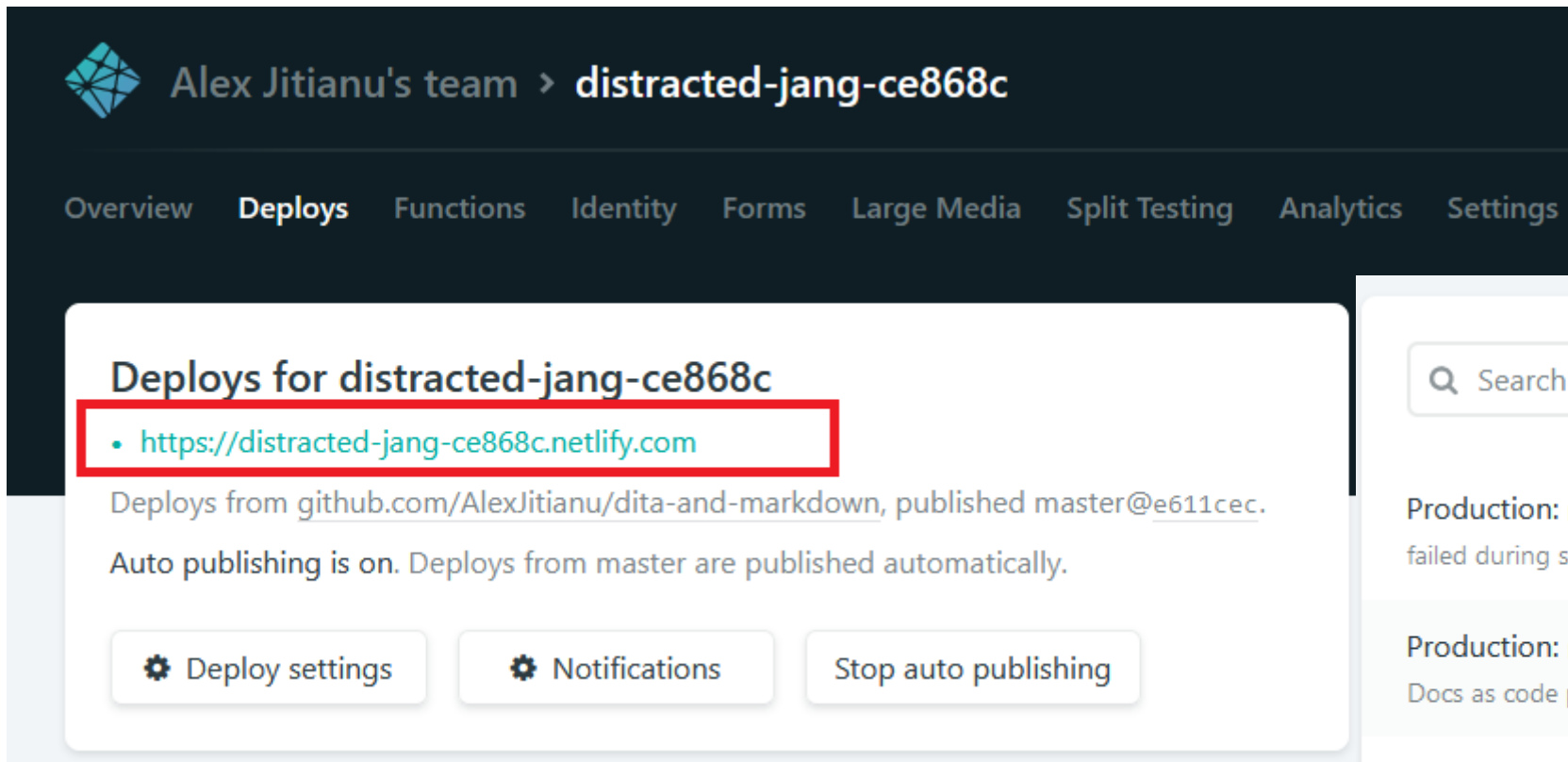
What can we automate for a documentation project?

- Quality checks
 - Business rules (Schematron, Vale)
 - Integrity checks (Validate and Check for Completeness)
- Reuse metrics
- Publishing pipelines

Proposed solution

- Text Markup: **DITA + Markdown**
- Version control: **GitHub**
- Issues tracker: **GitHub**
- CI/CD : **Netlify + SonarCloud**
- Collaboration: **Oxygen Web Author** [links in published output]

Netlify





Alex Jitianu's team > distracted-jang-ce868c

Overview **Deploys** Functions Identity Forms Large Media Split Testing Analytics Settings

Deploys for distracted-jang-ce868c

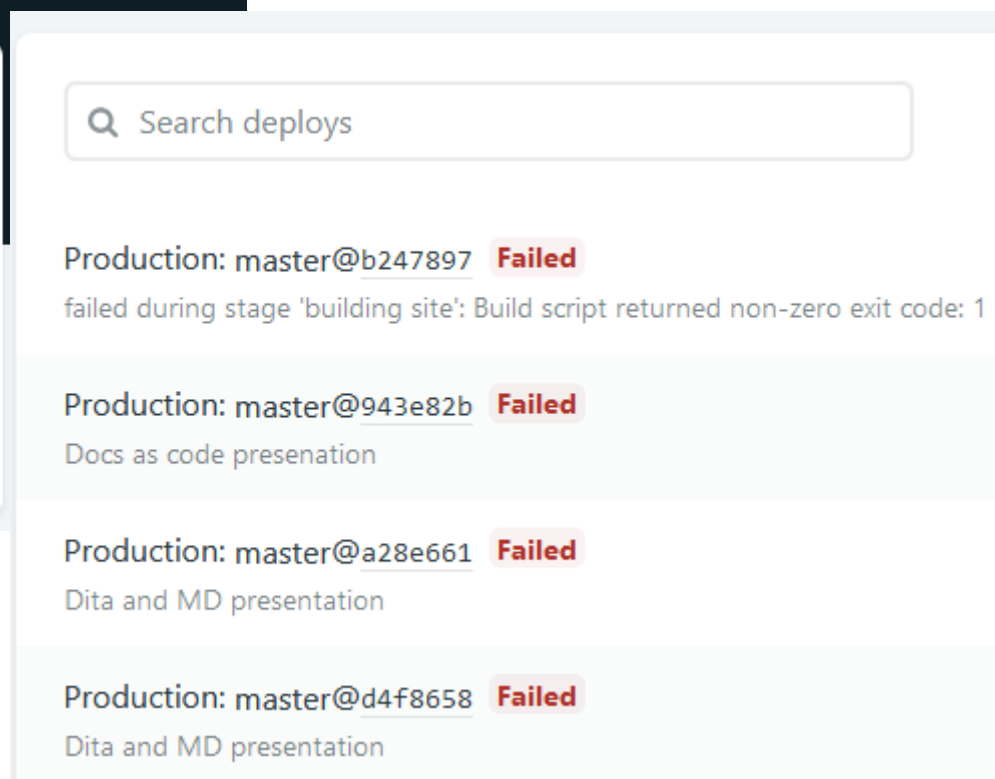
- <https://distracted-jang-ce868c.netlify.com>

Deploys from [github.com/AlexJitianu/dita-and-markdown](#), published master@[e611cec](#).
Auto publishing is on. Deploys from master are published automatically.

 Deploy settings  Notifications Stop auto publishing

A platform offering:

- Hosting (free!)
- Continuous Deployment from a Git Repo



Search deploys

Production: master@ b247897	Failed
failed during stage 'building site': Build script returned non-zero exit code: 1	
Production: master@ 943e82b	Failed
Docs as code presentation	
Production: master@ a28e661	Failed
Dita and MD presentation	
Production: master@ d4f8658	Failed
Dita and MD presentation	

SonarCloud

An open-source platform for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities.

sonarcloud

AJ Alex Jitianu / dita-and-markdown master

Overview Issues Measures Code Activity

▼ Rule

Search for rules...

Schematron:id
VCC:Key definition
VCC:Profile
VCC:Reference

4 shown

source/markdown-dita/concepts/lawnmower.dita

File not found: "/opt/build/repo/source/markdown-dita/image/lawn-mower-311862_640.png".

See Rule VCC

2 months ago L8

Code Smell Minor Open Not assigned 5min effort

No tags

source/markdown-dita/garage.ditamap

Condition "novice" set on the "audience" attribute can never be effective because it is not present in the list of conditions from the ancestors: "expert".

See Rule VCC

2 months ago L20

Code Smell Minor Open Alex Jitianu 5min effort

No tags

source/rule_based_validation_samples.dita

Unreferenced key definition: "carwash".

See Rule VCC

2 months ago L45

Code Smell Minor Open Alex Jitianu 5min effort

No tags

source/rule_based_validation_samples.dita

List items should not end with a semi-colon (;). If it is a sentence then end it with a full stop (.), otherwise leave it without an ending character.

See Rule SCHEMATRON

2 months ago L18

Code Smell Minor Open Alex Jitianu 5min effort

No tags

Hands-on time

- Deploy to Netlify our *docs-as-code* solution for DITA and Markdown
 - <https://github.com/AlexJitianu/dita-meets-markdown>



SonarCloud Configuration

These details need to be filled in on a per-project basis

sonar.organization={sonarcloud.organization.name}

sonar.login={sonarcloud.auth.token}

sonar.projectKey={unique.project.name} !!!!!

Configuration

sonar.sources=.

sonar.host.url=https://sonarcloud.io

sonar.exclusions=bin/**,scripts/**, demo-files/**

sonar.externalIssuesReportPaths=bin/tmp/sonar-schematron.json,bin/tmp-vcc/vcc-result-sonar.json

<https://sonarcloud.io/account/security/>


<https://sonarcloud.io/account/organizations>

Hands-on time

- Setup SonarCloud
 - Create account
 - <https://sonarcloud.io/about>
 - Create organization
 - <https://sonarcloud.io/account/organizations>
 - Set permissions (*Execute Analysis*)
 - <https://sonarcloud.io/organizations/{organization-name}/permissions>
 - Generate Token
 - <https://sonarcloud.io/account/security/>



Sonarcloud permissions

 [My Projects](#) [My Issues](#) NEW CircleCI support, new rules for ... [Explore](#)

AJ Alex Jitianu Alex Jitianu's personal organization


Key: alexjitianu-github

[Projects](#) [Issues](#) [Quality Profiles](#) [Rules](#) [Quality Gates](#) [Members](#) [Administration](#)

Permissions

Grant and revoke organization permissions. Permissions can be granted to groups or individual users.

All Users Groups

	Administer Organization ?	Administer ?	Execute Analysis ?	Create ?
 Alex Jitianu Alex.Jitianu@github	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Quality Gates <input checked="" type="checkbox"/> Quality Profiles	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/> Projects

THANK YOU!

Any questions?

Alex Jitianu

alex_jitianu@oxygenxml.com

[@AlexJitianu](#)