Docs as Code

Documentation Management Inspired by Software Development

Alex Jitianu

alex_jitianu@oxygenxml.com

@AlexJitianu

syncro soft

<oXygen/>® XML Editor

Agenda

- What is *Docs as Code*
- Choosing a text markup language
- Version Control
- Continuous integration and delivery
- Example of a working *Docs as Code* setup

Docs as Code

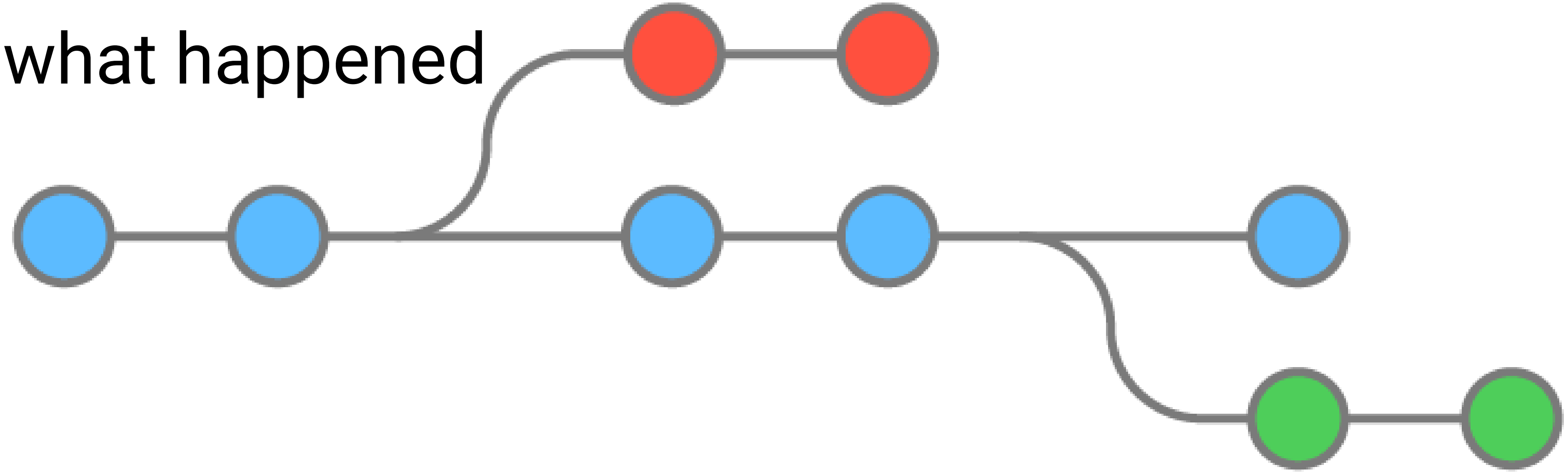
- Refers to a philosophy that you should be writing documentation with the same **tools** and **workflows** as code:
  - Version Control
  - Collaboration and Review Process
  - Automated Tests, Builds, Delivery
  - Issue Trackers
  - Plain Text Markup

Choosing a Suitable Text Markup

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

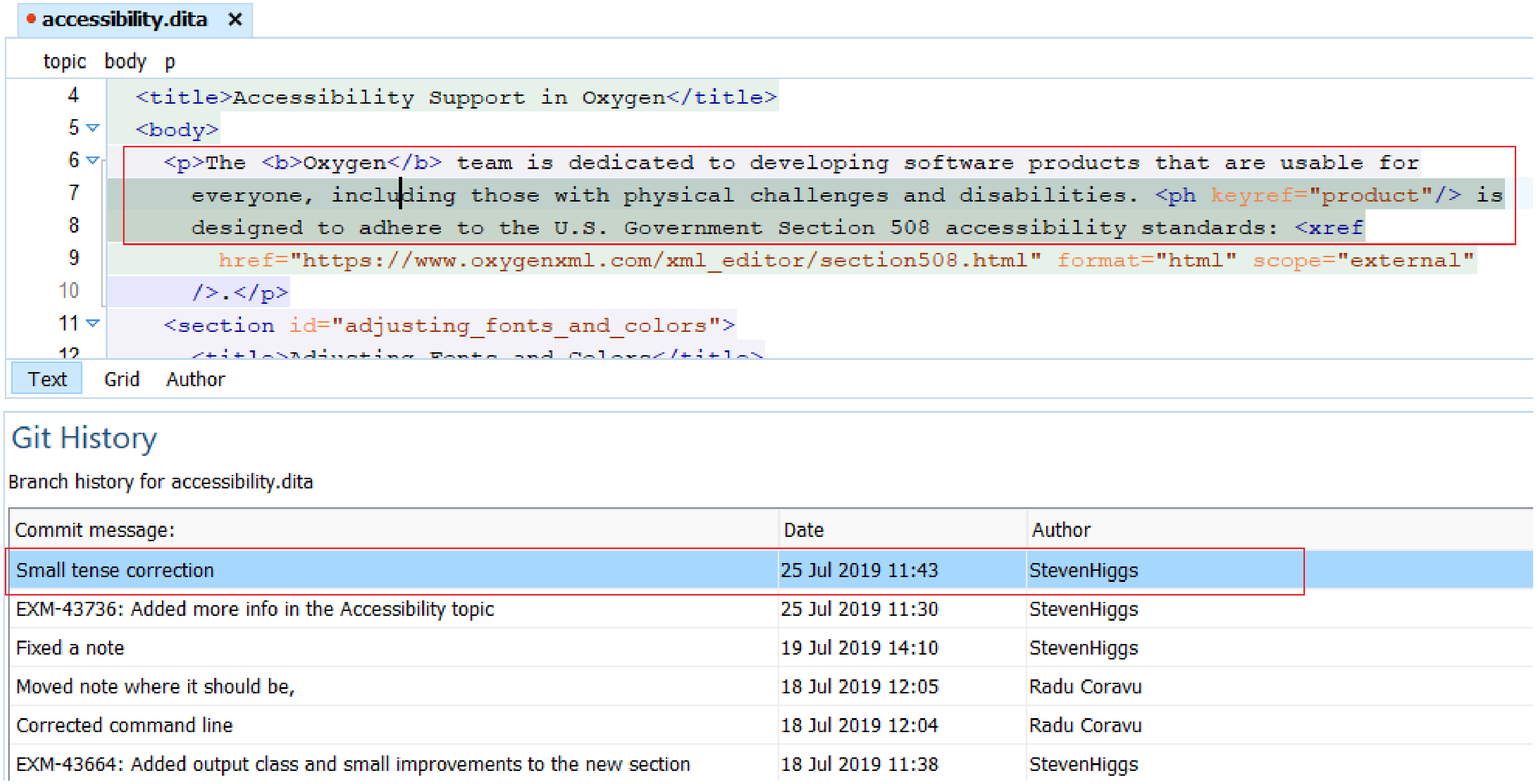
Why should I use a Version Control?

- Storing versions
  - *The basics of version control is the ability to save changes made to files, whilst retaining the changes from all previous versions.*
- Collaboration and review
- Understand what happened



Why should I use a Version Control?

- Understand what happened



```
<title>Accessibility Support in Oxygen</title>
<body>
  <p>The <b>Oxygen</b> team is dedicated to developing software products that are usable for
    everyone, including those with physical challenges and disabilities. <ph keyref="product"/> is
    designed to adhere to the U.S. Government Section 508 accessibility standards: <xref
      href="https://www.oxygenxml.com/xml_editor/section508.html" format="html" scope="external"
    />.</p>
  <section id="adjusting_fonts_and_colors">
    <title>Adjusting Fonts and Colors</title>
```

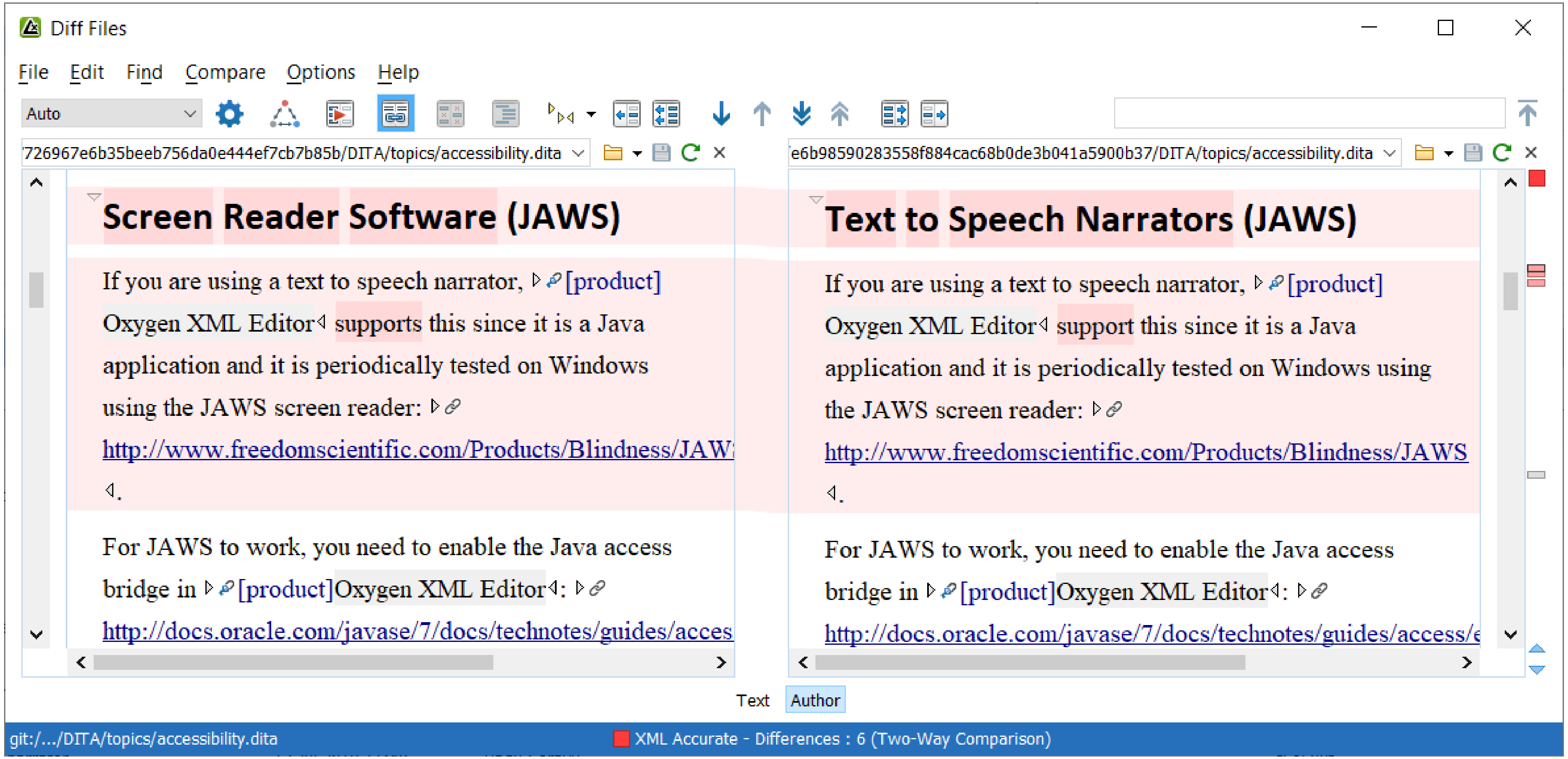
Git History

Branch history for accessibility.dita

Commit message:	Date	Author
Small tense correction	25 Jul 2019 11:43	StevenHiggs
EXM-43736: Added more info in the Accessibility topic	25 Jul 2019 11:30	StevenHiggs
Fixed a note	19 Jul 2019 14:10	StevenHiggs
Moved note where it should be,	18 Jul 2019 12:05	Radu Coravu
Corrected command line	18 Jul 2019 12:04	Radu Coravu
EXM-43664: Added output class and small improvements to the new section	18 Jul 2019 11:38	StevenHiggs

Why should I use a Version Control?

- Understand what happened



726967e6b35beeb756da0e444ef7cb7b85b/DITA/topics/accessibility.dita

e6b98590283558f884cac68b0de3b041a5900b37/DITA/topics/accessibility.dita

Screen Reader Software (JAWS)

If you are using a text to speech narrator, [product]Oxygen XML Editor supports this since it is a Java application and it is periodically tested on Windows using the JAWS screen reader: http://www.freedomscientific.com/Products/Blindness/JAW

For JAWS to work, you need to enable the Java access bridge in [product]Oxygen XML Editor: http://docs.oracle.com/javase/7/docs/technotes/guides/acces

Text to Speech Narrators (JAWS)

If you are using a text to speech narrator, [product]Oxygen XML Editor support this since it is a Java application and it is periodically tested on Windows using the JAWS screen reader: http://www.freedomscientific.com/Products/Blindness/JAWS.

For JAWS to work, you need to enable the Java access bridge in [product]Oxygen XML Editor: http://docs.oracle.com/javase/7/docs/technotes/guides/access/e

git:/.../DITA/topics/accessibility.dita

🟥 XML Accurate - Differences : 6 (Two-Way Comparison)

Which Version Control System?

- Git
- Mercurial
- Subversion

In the Cloud or on Premise

- On premise Git repositories
  - *GitLab Community Edition (CE) is an* ***open source*** *end-to-end software development platform with built-in version control, issue tracking, code review, CI/CD, and more. Self-host GitLab CE on your own servers, in a container, or on a cloud provider.*
- Web-based Git repositories
  - GitLab [https://about.gitlab.com/pricing]
  - GitHub [https://github.com/pricing]
  - Bitbucket [https://bitbucket.org/product/pricing]

Continuous Integration (CI)

- Changes are validated as soon as they are committed, by creating a build and running *automated tests*.
  - *Helps avoid “integration hell” where the software works on individual developers’ machines, but it fails when all developers combine (or “integrate”) their code*
- Puts a great emphasis on *testing automation.*

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

Continuous Delivery (CD)

- The goal of CD is to make sure the software is always ready to go to production
  - You have automated your release process
  - You can deploy your application at any point of time by clicking on a button.

CI/CD Platforms/Servers

- Jenkins
- Travis CI
- Netlify
- GitLab CI/CD

Editing tools

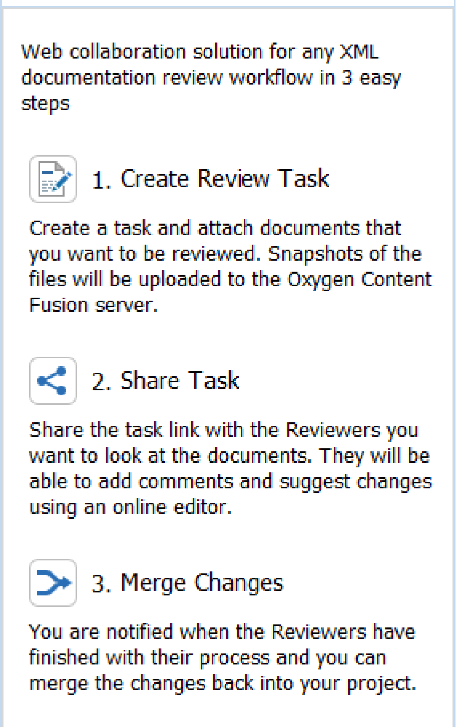
- Any text processor
- GitLab, GitHub built-in text editors
- Commercial XML editors
  - Oxygen Web Author
  - Oxygen XML Editor

Collaboration

- Using version control (Git)
- GitHub pull requests
  - https://github.com/features/code-review/
  - https://github.com/oxygenxml/blog/pull/25
- Using dedicated solutions (Oxygen Content Fusion, Oxygen XML Web Author)

DEMO TIME

- Oxygen Web Author in action
- Oxygen Content Fusion in action



Web collaboration solution for any XML documentation review workflow in 3 easy steps

1. Create Review Task

Create a task and attach documents that you want to be reviewed. Snapshots of the files will be uploaded to the Oxygen Content Fusion server.

2. Share Task

Share the task link with the Reviewers you want to look at the documents. They will be able to add comments and suggest changes using an online editor.

3. Merge Changes

You are notified when the Reviewers have finished with their process and you can merge the changes back into your project.

Issue Tracking

- GitHub Issue Tracker
  - https://github.com/oxygenxml/userguide/issues
- GitLab Issue Tracker
  - https://gitlab.com/jitianualex83/my-test-project/issues
- Atlassian Jira
  - https://www.atlassian.com/ro/software/jira

Proposed *Docs as Code* setup

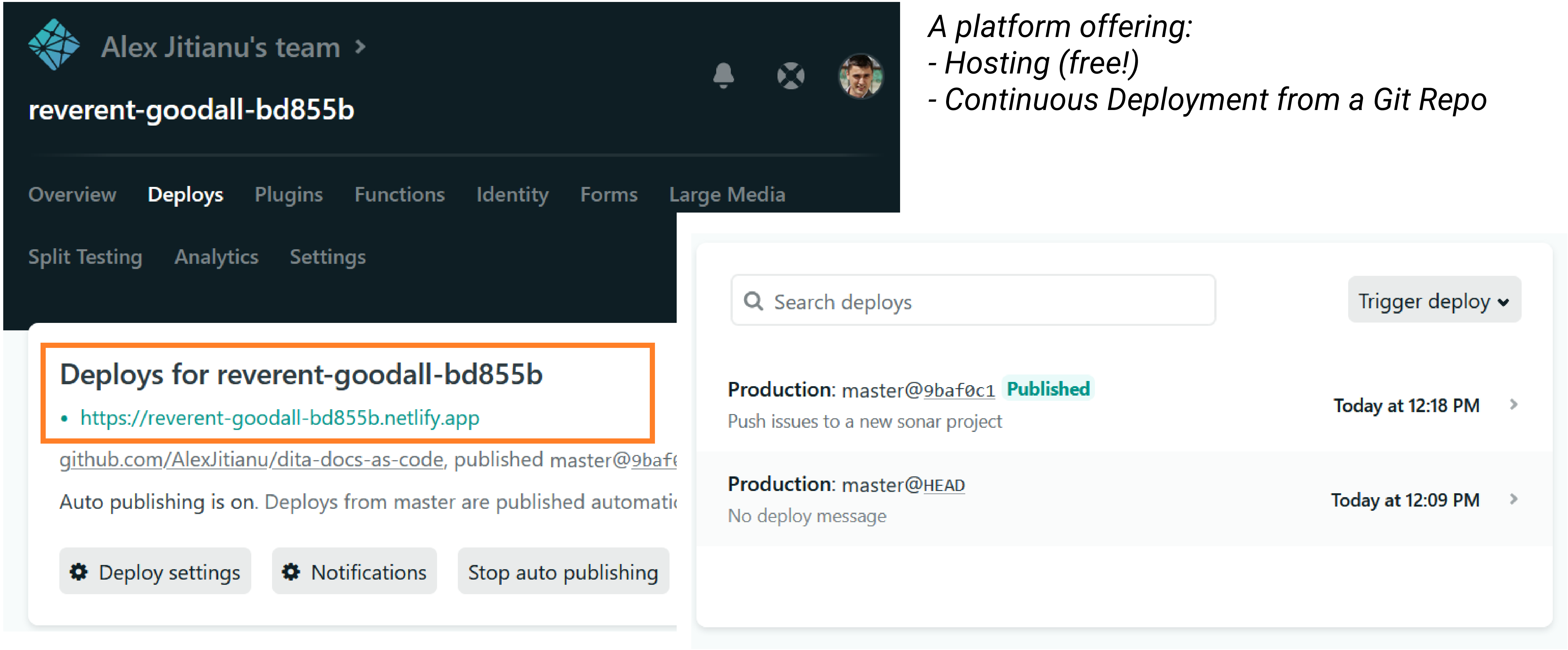
- Text Markup: *DITA + Markdown*
- Version control: *GitHub*
- Issues tracker: *GitHub*
- CI/CD : *Netlify + SonarCloud*
- Collaboration: *Oxygen Web Author* [links in published output]

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

Netlify

A platform offering:
- Hosting (free!)
- Continuous Deployment from a Git Repo



Alex Jitianu's team ›

reverent-goodall-bd855b

Deploys for reverent-goodall-bd855b

- https://reverent-goodall-bd855b.netlify.app

github.com/AlexJitianu/dita-docs-as-code, published master@9baf

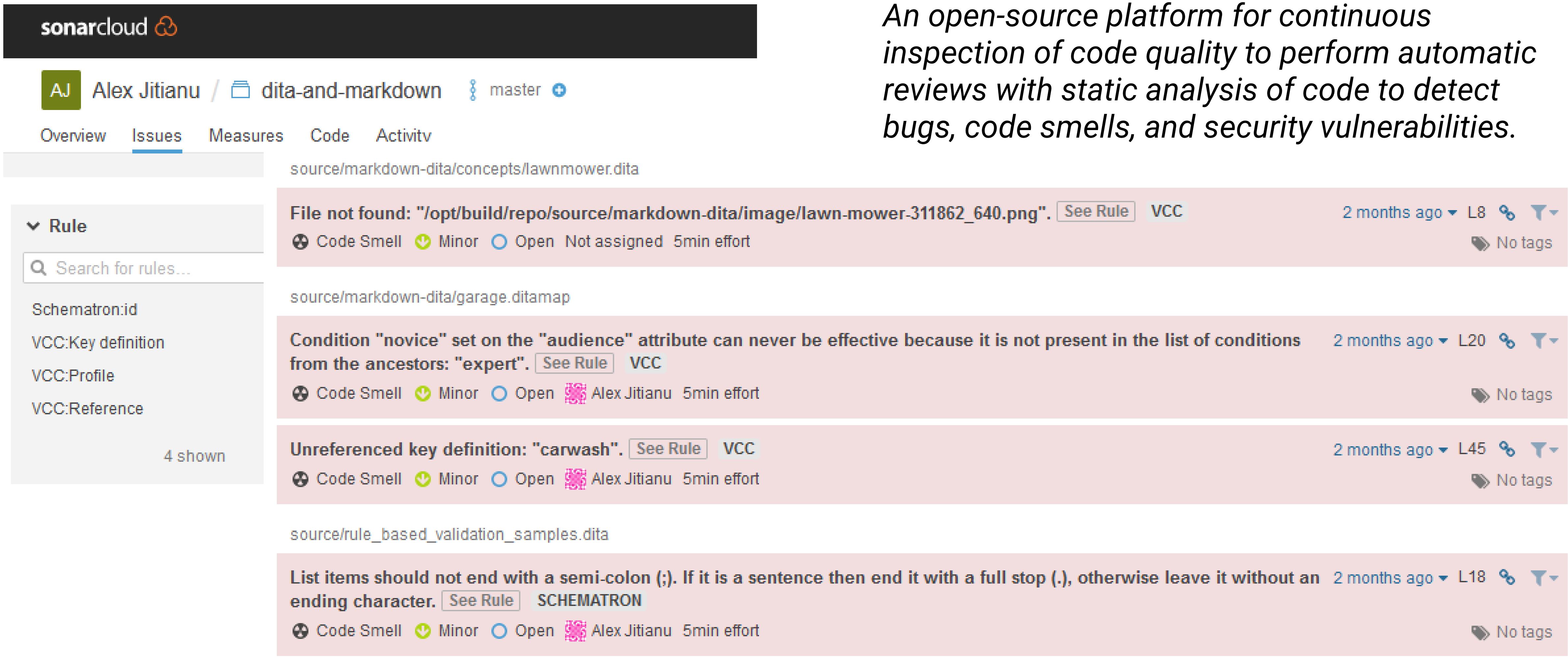
Auto publishing is on. Deploys from master are published automati

Production: master@9baf0c1 **Published**
Push issues to a new sonar project
Today at 12:18 PM

Production: master@HEAD
No deploy message
Today at 12:09 PM

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

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

DEMO TIME

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

Deploy to netlify

SonarCloud Configuration

- 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/

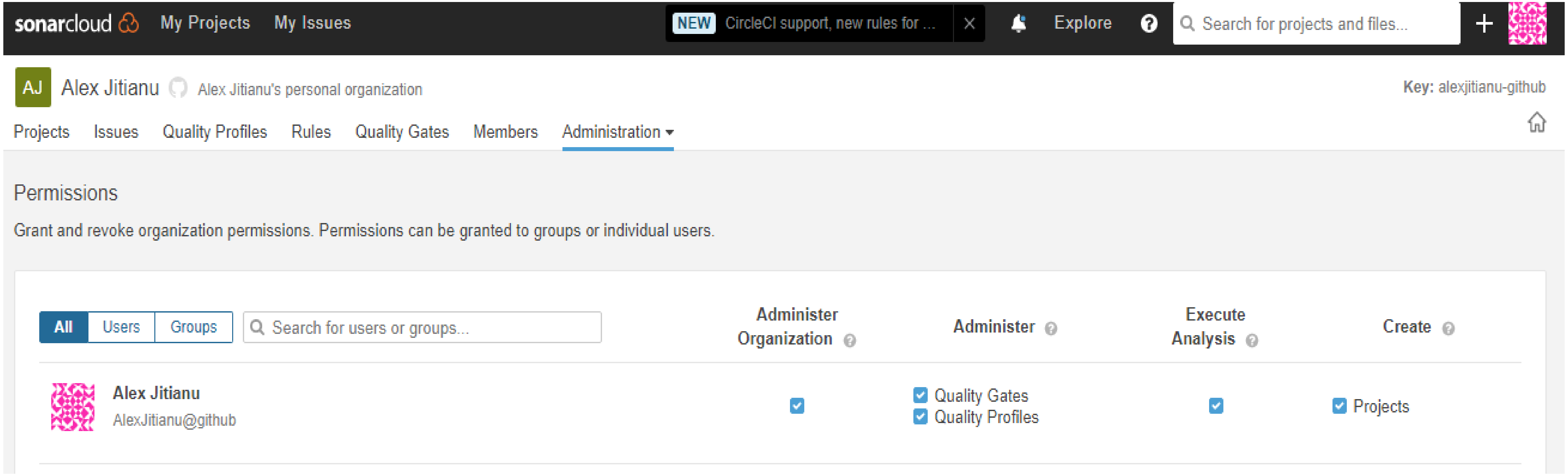
SonarCloud Configuration

```
# 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} !!!!!

# Configration
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://github.com/AlexJitianu/dita-docs-as-code/blob/master/scripts/sonar/sonar.properties

SonarCloud permissions



Permissions

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

	Administer Organization	Administer	Execute Analysis	Create
Alex Jitianu AlexJitianu@github	☑	☑ Quality Gates ☑ Quality Profiles	☑	☑ Projects

THANK YOU!

Alex Jitianu
alex_jitianu@oxygenxml.com
@AlexJitianu