



**If It's Not Documented, It Doesn't Exist:
Notes from the Write the Docs 2020
Conference**



September 10, 2020

Outline



1. How the virtual conference worked
2. Ideas from talks
 1. Craft of writing
 2. Technical details: pipelines, docs as code, single source
 3. Managing documentation projects; documentation debt
3. Future directions
4. Cannibalism

“Wherever you go, there you are.”



The screenshot shows a Zoom meeting interface. The main window displays a slide for "The biggest tech conference" hosted by Opers. The slide text includes: "The biggest tech conference", "So many creative people have never been gathered together.", "Jul 1, 9:00AM - Jul 2, 4:00PM UTC", and a "Stage is live now!" button with an "Enter stage" button. Below the slide is a "Description" section with text about a worldwide emergency and community value. The right sidebar shows a chat window with messages from participants like Joseph, Sara, Ronnie, Yesica, Miko (Organizer), Sofia, Maria, Ben, and Yesica. The chat window has a "Start typing..." input field at the bottom.

Opers
The biggest tech conference
So many creative people have never been gathered together.
Jul 1, 9:00AM - Jul 2, 4:00PM UTC

Time Left 12h:33m

The biggest tech conference
Hosted by Opers

Stage is live now! Enter stage

Description

We are facing a worldwide emergency that will deeply change the world and approach. We are all learning the immense lesson that this experience is teaching: be generous in what you do, how you do it, in the reason that moves you to do, work together to improve and not just for your own interest.

The value of a community is measured in moments like this.

The ability to be together with others, while staying inside the home, to inspire, improve and rediscover sharing as a key to our social life.

Chat | People

Event Messages

Joseph · 12:30
Hi everyone, great to be here!

Sara · 12:30
Hey @everyone, I have been waiting for this conference for three months!

Ronnie · 12:33
The day has finally come! 🎉

Yesica · 12:31
Do you know what time Elon starts his speech?

Miko - Organizer · 12:34
Hello! We are rounding up all preparations.

Sofia · 12:44
I have been waiting for this conference for three months!

Miko - Organizer · 12:45
ARE YOU READY?

Maria · 12:47
Hi! Let everyone have a lovely time today!

Ben · 12:50
Hi guys! Any folks from Chicago in here?

Yesica · 12:47
Yes! Let's get it started!

Start typing...

Benefits of Hopin platform



<https://hopin.to>

- Usability — entirely browser-based (using WebRTC)
- Flexibility — Keynote talks were pre-recorded, broadcast via YouTube
- Utility — “rooms” were useful in a variety of ways:
 - good analog for informal, ad-hoc discussions
 - Ability to go “into a corner” for one-on-one discussions
 - Easy to set up talks, sessions outside of keynotes
 - “Networking” feature that would pair you randomly with other users

Networking



- Many (most?) technical writers at the conference came from other careers
 - Software developers
 - Project managers
 - Journalists
 - . . . and one or two people with actual English degrees
- Seemed diverse — age, gender, race, sex
- There is a lot of literature out there (of course) about technical writing, as well as blogs, other resources
- WriteTheDocs Slack channel

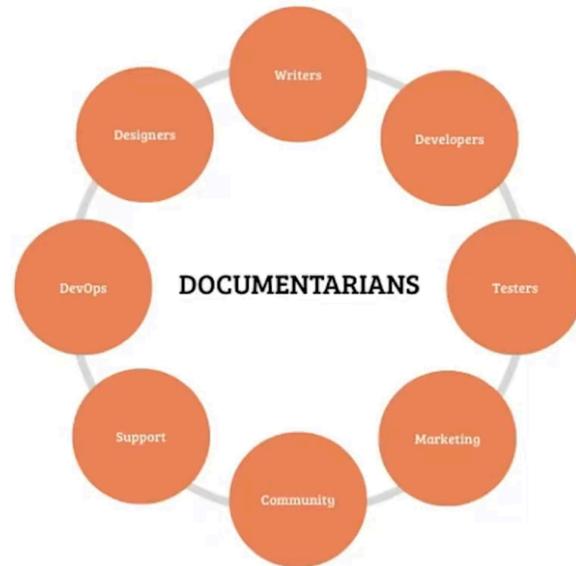
Documentarians!



Documentarians!



Documentarians!



Documentarians!



- Writers
- Developers
- Testers
- Marketing
- Community
- Support
- DevOps
- Designers
- Scientists, machinists, engineers, technicians
- Anyone that works with other people

Humans == external data storage



Humans == external data storage



- Craft of writing
 - Accessibility (a11y), internationalization (i18n)
 - Creating quality code samples
- Tools and Pipelines
 - Using Jupyter to both document and run tests
 - Workflows — single source, using pandoc, other tools
 - Creating docs from structured data (such as APIs)
- Management
 - Migrating to a new platform
 - Community building
 - “Documentation debt”

Craft of writing — accessibility



Craft of writing — internationalization



Using Jupyter to both document and run tests

- "exercise", "solution", and "testing" tags for different cells
- Different tags for tutorials, published example, and CI set
- Astronomer might write a Jupyter notebook that would both serve as a tutorial, and have built-in tests (hidden from the users) that would always ensure the tutorial worked and was current
- nbconvert (part of Jupyter) used to convert notebooks into Sphinx docs for non-interactive publication

Tools and pipelines — workflows



From a talk by Dan Gunter at LBNL describing their pipeline:

- Sphinx uses a LaTeX (XeTeX/etc.) backend for generating PDF documentation alongside the static site generator
- Container setup with all the dependencies, LaTeX stuff, fonts, etc. has helped make our build pipelines very hands-off
- When content is checked in:
 - Site is generated
 - PDF is generated
 - Site is published

Tools and pipelines — workflows



- We're doing most of this at <https://docs.greenbankobservatory.org>
- Check-ins of code to the SDDdocs git repo fire off a build via Jenkins
- Deploys new version if successful
- Also automatically grabs from docs/ directories of various repositories
 - Still need to add automatic PDF generation
 - Organization improvements also needed

Linting with vale



```
ls
test.md

vale .

test.md
8:93    warning  Avoid hyphens in ages unless      18F.Ages
        it clarifies the text.
14:16   error    Use 'DC' instead of 'D.C.'        18F.Abbreviations
17:162  warning  Prefer 'drop-down' or 'drop      18F.Terms
        down' over 'dropdown.'
45:18   error    Use 'U.S.' instead of 'US'        18F.Abbreviations
45:26   error    Use 'U.S.' instead of 'USA'       18F.Abbreviations

* 3 errors, 2 warnings and 0 suggestions in 1 file.

~/go/src/github.com/errata-ai/vale/fixtures/styles/18F/  master
```

Documentation debt



- Analyze
- Design and structure
- Develop content
- Preview and feedback
- Publish
- Maintain

Takeaway: see documentation — even if it's just internal — as a product (or in our case, a project)

Documentation debt



1. **Scope, define, map the problem of doc debt:** make a problem statement concise and clear. What to include and what not?
2. **Research and validate assumptions**
3. **Write a product requirements doc:** why do you need doc refactoring? What does it entail? Who benefits? How does it benefit them?
4. **Create a strategy and roadmap:** what is the plan of action? Who is involved? What does everyone need to do?
5. **Kick-off meeting with all stakeholders:** Alignment! Everyone should be aware of the plan. Get feedback!
6. **Develop a proof of concept:** start with *something*. Test, test, test. Iterate!
7. **Develop the product:** build the product, build the solution
8. **Publish and maintain the product:** take it live. Monitor and keep track. Retrospective and document those lessons learned.

Grammar matters!



Grammar matters!



Resources



- WtD Portland 2020: <https://www.writethedocs.org/conf/portland/2020/schedule/>
- Talks: <https://www.youtube.com/playlist?list=PLZAeFn6dfHpkBJAPYFrob6gqdiBuePwGJ>
- Hopin: <https://hopin.to/>
- Vale: <https://github.com/apps/vale-linter>
- Writing to Learn: https://www.goodreads.com/book/show/585474.Writing_to_Learn
- Modern Technical Writing: An Introduction to Software Documentation: <https://www.goodreads.com/book/show/28433138-modern-technical-writing>



GREEN BANK OBSERVATORY

greenbankobservatory.org

*The Green Bank Observatory is a facility of the National Science Foundation
operated under cooperative agreement by Associated Universities, Inc.*