Please feel free to share and use these notes. Leave comments if I'm missing ideas, or misrepresented something!

Cheers,

Sarah
@sarahleeyoga

Postulating the Backlog Laxative

Paul Adams - @therealpadams

Scrum Process:

- Comes from rugby - everyone in the scrum is pushing the same way, but everyone in the scrum has a specific job (ie. the hooker to grab the ball)
- Scrum is all about delivering value - whatever you're doing now, should be the best thing to do now, and it should deliver high value first.
- Often introduced when you need to scale, and do more, better.

The problem: How does documentation fit into a scrum?

Some common pitfalls:

- Do nothing. Carry on, scale the engineering team, but don't bother with documentation.
- Make the fish ride a bicycle. Put technical writers into the scrum team.
    - This is pretty common.
    - Not very efficient
        - there's a communication overhead and tech writers become a bit of a distraction

        - You're spending time creating stories for just one person on the team which goes against the theory of scrum - a backlog where anyone on the team can pick up the next story.
    - Keep tech writers in the team, but they are out of sync by one sprint

Forget Scrum, think Rugby
- In Rugby, when you're in a scrum, there's 6 members of the team that **aren't involved**
- Technical writers don't need to be part of the Scrum team
- Scrum is not important, but being agile is.

SuperGran - the symbol of feminist empowerment in Scotland (#themoreyouknow)
- Scrum developed to empower people
- So how do we empower technical writers?

Try something else.

Enable the people who know how technical writing actually works. Don't make the be the fish on the bike.
- "You don't have to be in scrum, you just have to work smoothly with the engineering scrum team"
- Try and get tech writers to drive

Agility > Process
Maintain One Goal (but maybe many teams)
Empower and Enable teams

Writing as a non-native speaker

Istvan Zoltan Szabo - @szabosteve
steve@pronovix.com

Language is the most powerful tool natural selection has ever devised.

1. **The Limits of My Language**

- I am thinking something complex, but I don't have the language to convey it.
- "The limits of my language mean the limits of my world" - Ludwig Wittgenstein

2. **Entropy in communication**

- obscurity , uncertainty, ambiguity
- Constantly working against the transmission of information
    - In speech we use redundancy, but this is unnecessary in writing
- “If I had more time, I would have written a shorter letter”
- English lacks preciseness other languages have

3. How can we stretch the borders?

- Follow a process
    - Write - Create structure, estimate word count, focus on flow.
    - Let the draft rest - switch your mind to editing mode
    - Editing - Self edit, check grammar (with grammar checkers), read out loud, send to editor
- Ways to stretch the borders of your language
    - Translating a book (nonfiction is easier)
    - Watch a series - shows the most recent state of language, teaches pop culture (which finds its way into very different context!)
    - Get a mentor - not a teacher, but a coach. Their advice is more strategic than technical
        - This is difficult - but they come from personal or professional connections and there’s no shortcut
        - If no mentor - Get an Amazon account [READ], read blogs, go to events and meetups
    - Practice
        - keep a daily journal (helps to articulate thoughts quickly and effective, and you can track your progress)
        - Start a blog or Medium account
        - Create a workflow to stay on the path
        - Give a presentation at Write the Docs

4. How to survive the editing process
    - It’s always worse than it seems

Healthy Minds in a Healthy Community

Erik Romjin @erikpub

Code of Kindness

None of us are alone

- Sometimes you meet people who have all their shit together, they get along with everyone instantly, and life is amazing
- But actually - so many of these people have their own struggles

- I know very few people who haven't struggled with their own mental wellbeing
- No matter how successful someone seems, at some point they may be struggling just to get through a day

70% of people regularly experience physical symptoms due to stress
- "I'm fine I'm just tired" <- a common explanation

No matter what your struggle, there are other people in this room with similar struggles

GREAT idea - Djangocon Europe 2015, attendees were able to speak to a counselor. 1 in 10 attendees used the service.
- People felt listened to, validated and were able to talk things out

We're not mental health professionals, we can make a difference
- Be empathetic, listen
- We're not alone

Help Yourself
- People love contributing to projects etc, but we often forget that **we need to help ourselves before we help others.**
- Just like in an airplane - only after putting on your own mask can you help your dependants.
- It's ok to say NO and it's okay to say NO MORE to things that you've already committed to

Sustainability isn't just for Open Source, but also for the Open Sourcers.

Communicating your need for self help to others may seem like a big hurdle
- Remember: what you do != who you are.
- **Suffering through our work serves no-one.**
    - If you let your projects damage yourself, there's no point in doing them
- You can imagine that the people around you won't accept your choices
    - But in reality, people react with kindness and understanding

Asking for help is not the same as failing
- Staying in a role, or taking on a task that you're unable to complete is like licking a cookie - no one else can eat the cookie until you step back.
- It's actually The Opposite
    - Asking for help is succeeding!

Don't bury your head in the sand
- Even if you are successfully hiding your panic attacks, you are still having them

- When you bring issues out into daylight, you're not making them more real

Others do not know what you need if you do not ASK
- Eg. ask conference organizers to schedule you earlier because it reduces your anxiety
- They won't know to do it if you don't ask - but probably will help you if you do!

It's okay to ask for help, because we are a COMMUNITY

Build a culture that makes *everyone* feel like they are welcome and you're delighted to have them (specifically them) involved.
- Eg. Quiet Rooms where people can unwind
- Supportive posters welcoming people
- Provide information to find recovery meetings in the local area (for those out of town)
    - Shows that someone in the community understand and care about your struggles
- Assume an opt-in membership which renews every 6 months - you have to explicitly say you want to keep going - combats overcommitment
- Django fellowship program - allows one person to be paid for Django work instead of having to balance a full time job

Django software foundation wellbeing committee
- Peer support network that allows people to connect with others in the community that need to talk to someone who understands.
- Theory - you don't need to be a mental health professional to make someone feel like a happy little sushi roll

If you are struggling, not only are you not alone, but you are also not any less loved

We don't always let people know how much we care
- It can feel uncomfortable to appreciate people, but it always is well received

Happiness packets - send appreciation to people you are grateful for.

Lightning Talks

- Algolia - DocSearch for your docs
- A role for AI in Documentation - @kvantomme
    - Recommended Book - Inevitable
    - bit.ly/AI-DOCS
- Working from home - tips tricks hacks
- All about testing - Test the Docs - there's an Unconference coming!

- Technical Writing Metaphors - Milan - @MilaNavratil
- Translating in non-xml languages
- Is Tech Writer the right name? Why it's bad. Beth - @baitman
- Why Write Docs - @ericholscher
- Documenting the mighty dinosaur software - Vasily

Documenting Data Center Operations

Joan Wendt

(Find youtube video with Data Center tour - it's very cool!)

Why do you want to be an Operations tech writer?

- "I'm glad he wasn't at the interview, because I'm not sure I had an answer"

What's different about Ops Tech Writing?

We document:

- Construction site prep
- Equipment/component assembly
- Component installation
- Testing, maintenance, repair, decommissioning

Eg. If you worked for a utility company, you would go out to the field photographing and documenting the pipes. It's the same thing :)

Workflow

- Attend design meetings
- Work in lab to photograph prototypes
- Accompany engineer to first build at site
- Create draft, return to DC, field test the doc
- Publish production version

Value is functional quality - it's all about the clarity, because things can get very complex

- No superfluous intro text
- Avoid def and indef articles
- No focus on formatting

Above all - keep it concise

Photos instead of screenshots

- Each photo needs perfectly precise detail
- Needs bird eye, worm eye, front back, inside, outside, so lots of climbing, stooping and contorting

Challenges
- Macho culture - difficult to get knowledge out of them
- Strict security policies and require up to 20 levels of approvals
- Detail oriented REQUIRED - projects can cost hundreds of millions of dollars and bad docs can screw it up. Incorrect safety docs can actually cause death from voltage or heavy equipment

Pros
- Lots of travel
- The Engineers are amazing and have actual social skills :)
- Field testing docs is a dream come true
- You get to meet the end user face to face
- Standardized setups mean that you can troubleshoot across sites
- When techs collaborate, best practices rise to the top and become standard practices and make it into the docs (everyone competes to

Cons
- Tedium
- Can be physically demanding
- No glory - it's a cost center, not an income producing area
- STRESSFUL (she didn't say this, but people dying due to doc mistakes seems stressful)

Watch that tone!

Creating an information experience in the Atlassian Voice
Sarah Karp @skarpediem

"Creating meaningful interactions with users requires a strategic voice with a human tone."
Kate Kiefer Lee - MailChimp

Voice = personality
Tone = mood

Great examples
- Warby Parker
- MailChimp - the voice stays the same, but tone changes throughout docs

Why is Voice and Tone so difficult?

- It changes across scenarios (billing, using the product, etc)
    - Need to interpret how customers are feeling across each scenario
- Tone has a personal preference

You've got the power

- Understand and utilize the full power. Everyone needs to know why it's helpful.
- Brand identity
- Human connection
    - **Poor copy may not be the reason people leave, but they definitely can be the reason the stay.**
- Customer engagement
    - Can copy drive usage of a feature

Define your own voice

- Look through your company values, and see what stands out that should be in your content too
- Bold, Optimistic, Practical, with a wink.
- All tone guidelines have an opposite too (eg. Bold, but not aggressive)

Voice doesn't change

- Document doesn't need to be identical, but it should stay consistent

Learn by example

- Internal repository of examples where "with a wink" landed well, or worked badly.

Voicify cards

- Have guidelines for different scenarios
- Eg. Error Messaging
    - Use less wink
    - Give customers next steps
    - Be specific
    - Etc.

https://twitter.com/baitman/status/777860407721652224

Voicify your content in 3 steps

- Set the scene
- Grab a card
- Review

Share your voicify powers

- Content audits
- Writing workshops - eg. bootcamp for new hires

Voice and tone builds trust - people know you and can identify you!

When bad screenshots happen to good writers

Swapnil Ogale @swapnilogale

People don't understand …
- That you don't need a screenshot for every step
- That screenshots need to be tailored for the environment

Fallouts
- Bad screenshots mean taking the time to go back and fix it

How to help
- Add screenshots into your Style Guides
    - What areas to snapshot
    - What format to save in
    - Post processing - what tools to use?
- Create a cheat sheet

Screenshots are most effective when they are meant to serve a specific purpose.

Information microarchitecture

Grammar, syntax, and cognitive rhetoric
Rory Tanner - @roringtonj

We help people understand things more deeply by using organization.

In 2015, users spent DECADES reading Shopify documents
- Engagement was happening at a level where incremental improvements were going to be beneficial
- "Editing is just refactoring code designed to run on someone else's brain" @mlms
- Language is code running in someone else's brain, so performance matters

How does reading work?
- Syntax is important
- Ambiguity really slows down processing time
- Eg. The defendant examined by the lawyer was unreliable
    - When we read it, we think the defendant was doing the examining… but the brain has to change the interpretation at the end of the sentence

    - “Temporal ambiguity”
- So, what syntax patterns (or IA architecture) is prone to this kind of ambiguity

If you’re dealing in spoken language, you have tone queues. But this doesn’t exist in written word.

One syntax pattern that does introduce ambiguity is the “If this, then this”.
“If you need help, read the docs”

It should be less ambiguous if you add a “then” to the sentence structure

“If you need help, then read the docs.”

Let’s test:

How does performance change when you add “then”? OR How can you tell if performance changes when you add “then”?

- Use Optimizely to run an A/B test where A versions have “thens” and B versions don’t
- Compare reader engagement between the two (theory - more engagement would happen from people who are less exhausted by ambiguity)
- Across 4 tests, we had improvements ranging from 0.6 - 2%
    - Might not be significant, but at large scales it’s interesting enough to keep looking into

Takeaways:

- Every sentence has an user journey.
- It’s polar bears all the way down - IA doesn’t just matter at the BIG level, it matters at the micro level too.

What IKEA and game manuals taught me about technical writing

@chrischinch

“Be curious and explore other things”

A board game is a series of steps that needs to be understood and followed by the gamer.

- It involves a lot more math than you think to make it work logically

Lessons learned

- Who are you writing for?

    - Ex. experienced gamers might not need to be told everything. But if you're trying to be more welcoming to newbies, you might need to be more explicit.
- Less can be more
    - Eg. IKEA parts manual - you don't need to know what they are, or what they do, but you just need the picture. No presumed knowledge
- Consider the context of the reader
    - Images, videos
- Consistent bite sized concepts
- Don't alienate the reader with concepts that make them feel stupid

Story telling
- Consistent examples and have fun

Documentoring

Growing a "Love the Docs" community
@DaveOliver79

IBM - 1 writer to 10 engineers
Startup - 1 writer to 25 engineers
Current company - 1 writer to 50 engineers

Being an effective curator can be more valuable than being a prolific creator
- Automation
- testing

The most qualified person to write software documentation is usually the software engineer
- As a TW, you spend so much of your time trying to get that information out of their head

The least appropriate person to write software documentation is usually the software engineer
- Their focus is usually very narrow, whereas a TW focus is one user, one use case

Great product/ flawed docs
- Lots of early interest
- Installation and configuration was difficult
- User feedback all revolved around the difficulty of the product
- Large proportion of product features not utilised by users
    - Sad, because they thought they had cutting edge features, but users weren't even getting that far
- Poor user retention

"Full props for technical ability, but you need to add documented to your definition of done. If you'd user tested your documentation, you'd have found a lot of these flaws earlier"

BUT, the local version of the product had easy installation, and simple configuration. Users got into all the features, and their feedback centered around improvements.

- This helped us realize that we didn't nail the documentation on the first product, and it had far reaching implications

Give yourself the best chance to succeed.

- Tooling - like JIRA, markdown, github
- 80% correct is better than 80% complete

Documentation should not be an island

- It should be a bridge

The documentoring "secret" -

- Persistence?
- Process?
- Events?
    - Paperjam - a whole day of feedback and updates on the docs
- Tantrums?
- Donuts? Baked goods?
- Empathy.

Pretty Hurts

Why Better is better that Best
Riona @rionam

Quality - what is it?

eg. Hiring question

- once you've done the docs, how do you know they are good?
- we don't have an established vocabulary to talk about quality of docs

We need to establish a vocabulary for documentation quality

- Challenging because we all work on different types of documents

Structural and Functional Quality
"I know it when I see it" ←- not enough

Structural

- Are the spelling and grammar correct?
- Style and usage guidelines
- Does it use proper voice and tone
- Is it well-organized?
- Is it easy to navigate?

Structural quality is “pretty” but it doesn’t measure effectiveness

Functional quality
- Does it do what it’s supposed to do?
- Does it satisfy requirements?
- Achieve what it sets out to achieve for users?

Eg. One that promotes product adoption.

High structural quality + Low functional quality = POOR overall quality

Functional Quality is ALWAYS the most important

Understand
Define
Execute
Measure

1. Start with functional requirements
    a. State what the document should achieve or what it’s supposed to DO
2. Define functional requirements with internal and external stakeholders
    a. Eg. deflect support calls, increase signups
    b. You can have all sorts of requirements, but the important thing is to acknowledge the
3. Execute on the docs!
4. Measure success
    a. To demonstrate value we need to focus on functional quality data
    b. Structural data is unpersuasive - “uses active voice 85% of the time” - meh.
    c. Sentiment data - are the stakeholders (including users) are happy with docs
        i. Use support data, ask users, etc
5. Use quality data to communicate impact and value
    a. Talking about impact with other teams
    b. Product teams and managers
    c. Eg. “We wanted to increase feature adoption by 10% in Q4”
        i. I restructured the docs
        ii. I worked with marketing for messaging to drive users to the docs
        iii. Follow up study

CTA - move away from defining our role as structural quality only
As we do so as writers, we will find ourselves better aligned with our teams and their goals

Checklist the Docs

Daniel Beck @ddbeck

Too complex to fly

- Runway was not littered with charred corpses of previously failed flights

Increasing complexity while maintaining reliability is not assured

We live in a world where even the best people will make mistakes - not because we are bad, but because systems become more complex and expectations increase.

Instead of blaming pilots for failures, Boeing decided to make it easier for pilots to manage the complexity ---> The Checklist

Now, every aviation company using checklists in some way.

The Checklist

- Completable, not aspirational
- Repeatable, not a to-do list
- You can re-use the checklist in similar situations

Find a good context

- Physical thresholds
- Routine pauses - like at the start of every skype call when your colleague says “let me restart my computer”
- Process boundaries - like a handoff

Choose a pattern

- Read-do - read it, then do the thing
- Do-confirm - do the work, and then go through the checklist to make sure everything is where you want it to be

Choose tasks

- Standard Procedure
- Essential Tasks

Use the checklist

- Exploit existing habits
- Reward yourself
- Log your progress

Finish

- Use physical checklists
- Add movement and sound
    - Eg. Pilots “point and shoot” - point at the item in the list, then point at the item they are checking
- Invent a ritual

Review the checklist

Checklists are really helpful for ensuring your Automations are working

- Verify
- Integrate - you can have git prompt you to run through a checklist
- Prototype

Recommended Read -
https://www.amazon.com/Checklist-Manifesto-How-Things-Right/dp/0312430000
“Unless you have surgery planned - then you should probably wait”

Checklists can feel like they reduce spontaneity or creativity - but I guarantee there are tasks that do not require those (ie. spontaneous deployments or creative backup systems)

Get the slides: ddbeck.com/wtdeu2016

<<Sarah took a mental break for a couple hours>>

What Writing Fiction will teach you about writing Documentation

Thursday Bram @thursdayb

http://www.thursdaybram.com/what-writing-fiction-will-teach-you-about-writing-documentation

Lightning Talks

Using Meaningful Names to improve API Documentation

Jan Christian Kraus

For good API documentation

- Include good examples
- Be complete
- Support many complex usage scenarios
- Be conveniently organized

Be Complete

Names allow Verification

Names can help detect gaps in API documentation by applying documentation patterns.

- Select a verb
- Apply the right pattern

Poll the Docs

Kata Nagygyorgy
@nagygyorgyKata

Docs as Code: The Missing Manual

Margaret Eker
Jennifer Rondeau

- Docs should live close to the code
- They should be iterative - just like code

Release Planning

1. Scope - is it an update? A full feature release? A new product?
2. Design - what are your deliverables? How will it be deployed?
3. Development - Write, Review, Test (just like code!)

Docs as code makes sense, because if you're all working in the same environment, developers can literally write the first draft.