Why Documentation Matters

Mike Nielson

BAD Camp 2018

Who am I?

Mike Nielson

Sr. Web Developer @ Red Hat

Working with Drupal since v4

Open Source evangelist

Foodie

Portlander since 2010

Who are you?

1. Developer
2. PM
3. Architect
4. **User**

https://commons.wikimedia.org/wiki/File:Edward_Charles_Pickering%27s_Harem_13_May_1913.jpg

What we're going to cover

- Why documentation is hard for people & projects
  - How documentation speaks to the overall care & vision of a project
- How to argue to expanding time spent creating documentation
  - It's not about the bus plan

What we're going to cover (pt. 2)

- How to make documentation fun
- What makes documentation good
  - How to improve documentation
- What makes documentation bad

Why documentation?

Documentation is necessary

https://twitter.com/CodeWisdom/status/849295255912534018

Writing documentation to (re) learn

Wanna get something permanently stuck in your head?

Write a wiki page about it.

Being stuck sucks

Getting unstuck feels

Ah-MAZ-ing!

Enough about me tho...

I write docs in the hope that my struggles aren't yours

We build on the progress made by others

Documentation is good for the team

- Brain dump
- Billable hours
- Fewer meetings

Documentation is great for the team

- Mind state
- Fewer trivial issues
- More brains grasp what's going on in the code

Q: Why do we document?

A: Because it's good

1. Good for ourselves
2. Good for the team
3. Good for the community

Why is documentation hard?

Why documentation sucks

- Hard to find
- Opaque
- Out of date

It’s complicated

https://twitter.com/nnja/status/1040000240189726721

Why documentation sucks

- Writing code ≠ writing documentation
  - Not even close

Makers aren't users

"Users of a design system often have different goals and motivations from the design system makers.

The minutiae of the design decisions that go into a component might be interesting or helpful for the makers of the system, but users may very well not care"

-- Brad Frost

http://bradfrost.com/blog/post/how-much-documentation-to-include-in-a-style-guide

Reading != Understanding

Understanding is the ultimate goal for people.

When a developer ***understands*** the code that goes into the project, the solutions that result are far, far better than the inverse.

Documentation ↠ Understanding

When a developer ***understands*** the code that goes into the project, the solutions that result are far, far better than the inverse.

The end goal of documentation is to plant the seed of understanding

Some concepts are hard to put into writing

Rhetorical exercise:

Produce an algorithm wherein you instruct:

- A 4-year old in the fine art of paper airplane construction
- How to tie a bow tie

CAVEAT - No images allowed! Words only

Let's pause here

Go ahead & take a minute to think about it.

- A 4-year old in the fine art of paper airplane construction
- How to tie a bow tie

Why documentation sucks

- Don't know what we don't know

- i.e. Outside context problem

https://commons.wikimedia.org/wiki/File:Defense.gov_News_Photo_020221-D-9880W-080.jpg

We don’t know what we don’t know

An Outside Context Problem was the sort of thing most civilisations encountered just once, and which they tended to encounter rather in the same way a sentence encountered a full stop.

-- Iain M Banks Excession

Poor language

Translating requirements into code is hard enough

Translating code *back* into meaningful intent

Don’t despair

This isn’t just a software problem.

Useful knowledge transfer is a whole other field - Education

Improving on the concepts undergirding education - Epistemology

We’re not alone

Written communication is inadequate to express some concepts .

Like car crashes.

Code has its advantages

Our languages are simpler

Our concepts less nuanced

https://commons.wikimedia.org/wiki/File:Ida_Rhodes_at_NBS_001.jpg

Make an argument to expand time spent documenting

Convince your Manager

(YMMV)

Getting buy in from the team,

especially with stakeholders

Can't. Be. Underestimated.

In it for the long run

Good:

- Uses logic, self interest ($$)
- Long term change
- Frequent

Bad:

- Short term
- Infrequent
- Manipulative
- Appeal to emotion (fear)

Bus plan

- What happens to the project if <NAME> gets hit by a bus?
  - Pros:
    - Visceral
    - Highly motivational
    - E.g. <NAME> gets strep throat

Bus plan

commons.wikimedia.org/wiki/File:Muni_2_Clement_bus_front.JPG

- What happens to the project if <NAME> gets hit by a bus?
  - Cons:
    - Short duration of effectiveness
    - One time use
    - Management by crisis

Faster onboarding

Poll: Time to first commit?

A. 6 hours?
B. 12 hours?
C. 32 hours?
D. 60 hours?

Save $$

Long term investment in the project

- Pros:
  - 100% true
  - Persuasive
- Cons:
  - Indefinite time horizon
  - Falls apart easily

If a community, project...

- Bigger project
- More mature community
- Welcoming to new users

HUGE
welcome mat

Lower training costs

No need to fly folks in to train a team - save the $$ build a set of good docs.

Train via video conference one afternoon & offer a weekly work session

Make documentation fun

Documentation ... fun?

Hear me out

Creating documentation doesn't have to be a drag

The following are suggestions

1. To make consuming docs better
2. To make creating docs more enjoyable

Have a documentation sprint

For one sprint,

Do nothing but write documentation.

Not one line of code.

SRSLY

Get the whole team involved

https://www.pexels.com/photo/woman-looking-at-stick-notes-1520144

- Get food & beverages
- Have hourly stretch breaks!
- Get your PM to review PRs

Documentation by team building

60 mins of onsite meeting

Pair off & write documentation for 30 mins

10 min mingle / restroom break

Review other team's docs for 10 mins & submit action items

10 min to revise & submit documentation

Got a whole week on-site?

Have 2 sessions!

Book end the week's work

With two hours writing documentation instead of one

Break it up

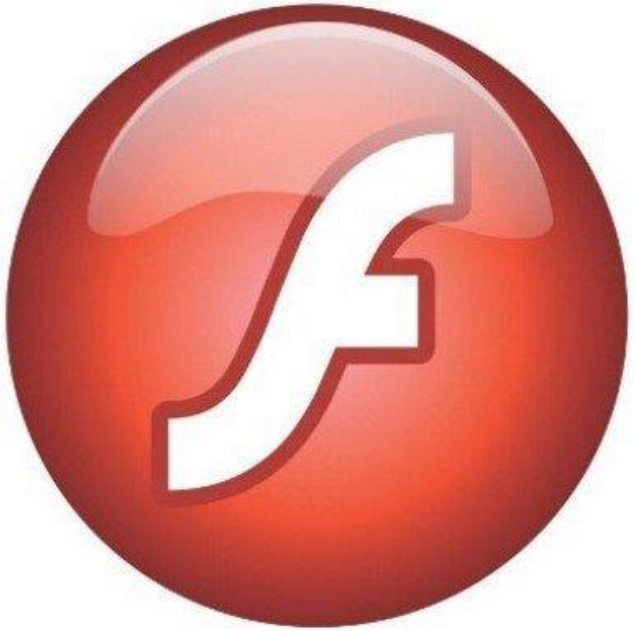
Add paragraph breaks

They’re a vacation for your eyes.

Add flash!

Not **that** kind of flash

- Images
- Gifs
- Emojis (where possible)



Add Photos

OSX :

Select area - cmd + shift + 4 (+ space)

Fullscreen - cmd + shift + 3

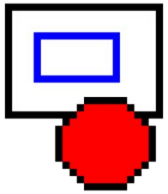
Linux: Screenshot

Windows: windows + shift + s or print scrn

Tools for going multimedia

Adding photos to documentation is easier than you think:

Licecap - make a .gif



Quicktime - audio & video

.gif FTW

Let's construct a query for such a component:

```
ery Hero($episode: Episode, $withFriends: Boo
nero(episode: $episode) {
  name
  friends @include(if: $withFriends) {
    name
  }
  __typename @skip(if: $withFriends)
}
```

VARIABLES

```
{
  "episode": "JEDI",
  "withFriends": false
}
```

```
{
  "data": {
    "hero": {
      "name": "R2-D2",
      "__typename": "Droid"
    }
  }
}
```

Try editing the variables above to instead pass `true` for `withFriends`, and see how the result changes.

Make documentation good

Documentation backlog

1. Create 1-2 documentation stories
    a. Break it into discrete chunks
    b. Assign it to someone (dev or otherwise)

2. Takes some time, write docs on their selected stories

Documentation backlog (cont.)

3. Go through the approval process

4. Ship it!

5. Iterate over the project

Create an index

Automate it, if you can

If you can't - make it single page & exhaustive

Encourage others to contribute to it

Non-technical folks edit

Docs should be understandable by anyone involved with the project

Add those unfamiliar with the nitty gritty details

An eye towards unfamiliarity with the implementation

Refactoring documentation

- Create tasks in template to document logic
  a. Refactor subtask!
- If you touch it, improve it
  a. At least add a TODO to come back & do more

Begin at the beginning

Encourage new hires to contribute to the docs as part of the onboarding

Set aside time during 1-on-1s to discuss their progress

Note down tricky spots & hangups

Better & better

Documentation requires revision.

Maybe something's changed

OR

Maybe there's more time to expand the scope

Mind the gap

What's trivial conclusions to a senior developer, may be a curveball to someone new

Focus on logical jumps that *appear* to be simple

Documentation as acceptance criteria

Mandate docs creation as part of marking off something as completed.

Maybe write the docs as a first pass at planning for implementation

Make documentation bad

Don’t do it

At all

Make it wrong

Bad documentation is worse than no documentation.

https://twitter.com/sodakpb/status/992007110501326849

Make a half-assed effort of it

0.o

Ehh... sure...

Does stuff...

Don’t make it easy to find

The README is three directories deep in `./not/my/problem/`

Want to read more?

Yeah, it’s on a pwd protected wiki with no index or structure

Violate Wheaton's law

// Simply do these (15) easy steps

// Follow the directions

** See **Mind the gap** **

No code as docs!

It's lazy & shows a lack of care to non-technical users

Code as documentation

“It’s easy enough to understand, just read the code”

WRONG!!1!!!

Wrapping up...

Wrapping up

In conclusion

- Writing for humans is hard
- Empathy is key
- Maintain the docs to avoid rot
- Make it fun!

To conclude

1. Documentation is good
    a. for you
    b. for the team
2. Do more of it
    a. Pitch it to your mgr
    b. Onboarding via docs
3. Avoid the pitfalls
    a. Comments != docs
    b. Mind the gap
4. Update as needed
    a. Docs sprint!

Interested? Let's talk!

about.me/nielsonm

Twitter: twitter.com/oswebguy

Github: github.com/nielsonm

LinkedIn: linkedin.com/in/nielsonm

Questions???

bit.ly/badcamp-write-the-docs

Thanks everyone!

bit.ly/badcamp-write-the-docs