Documentation: get it right!

ApacheCon Europe 2009
Amsterdam

Niels van Kampenhout
"Doctator"
Hippo
Amsterdam/San Francisco

About me

- My name is Niels van Kampenhout
- I work for Hippo
- My responsibilities include documentation and training
- I am a typical “user“ within Apache

This talk

- A user's perspective on documentation within Apache
- General observations
- How I think documentation should be done
- Not a writing course

Me and documentation

- I'm not really a technical writer
- I kind of became one by chance
- Common sense, trial and error
- I train people to use Apache software
    - I see how these people approach open source software

Open source and documentation

- Bad reputation
- It may not be **that** bad...
- ... but there is definitely room for improvement!

Why good documentation is important

- Adoption
- "Non open source people" have certain expectations

Documentation within Apache

- Is it really that bad?
- There is a lot of documentation, but...
  - Focus on developers
  - Inconsistent
  - Fragmented
- Folks outside Apache expect something more coherent, more guiding

The excuses

- It's open source, read the code!
- I don't have time to write documentation!
- I hate Xdoc/JSPWiki/...
- We have documentation, stop complaining!
- ~~We won't sell any training courses!~~

The real reasons

- Developers are generally not good documentation writers
- They're deep in code, find it difficult to step back to get an overview
- They're perfectionists
- Their talent is in coding, not in writing documentation

The real reasons

- Writing documentation is not sexy
- Why write documentation if you can fix a bug or implement a new feature?

The real reasons

- Fragmentation
  - Mailing lists
  - Wikis
  - Subprojects
- For an outsider it is hard to get an overview

What can we do about it?

- Take a step back

Be realistic

- Don't expect developers to write good documentation

Divide and conquer

- Make one person responsible for documentation (the "doctator")
- Doctator has overview, organizes
- Relieve (other) developers from **writing**, just let them **write down**

Embrace your users

- Users are often the best people to write user documentation
- They know what they want to achieve with your software
- There's a whole user list of them!

Embrace your users

- Make use of your community
- Encourage users to contribute documentation
- Guide documentation writing users towards committership

Embrace your users

- Have the right attitude towards your users
- “RTFM“ ?
  - Review The F***ing Manual!
- “We do this in our own time“
  - Users do it in their own time too you know!

Organizing your documentation

- Think about who will use your software, and what they want to achieve with it
- Write for a target audience
- Write to let the reader achieve something
- Validate your assumptions, ask your users!

Cater to your users

- Provide different tracks for different users and different goals
- Think about what to put in a track ...
- ... and what to leave out
- Tracks can share content
- But be careful not to let a user end up in a different track

Provide introductory documentation

- Introduce the reader to your project
- Describe in a few sentences what it is
- State what problem it is meant to solve
- Try to avoid acronyms, jargon
- A picture is worth a thousand words

Give your readers some context

- What do they need to know?
- What are they going to achieve?
- Where should they go next?
- Don't expect your readers to start on the homepage / table of contents

Respect your readers

- Your readers are smart people
- Don't force them to read what they don't need to know
- Don't talk to them as if they're children
- Imagine yourself in the reader's place

Think about the Big Picture

- Don't only look at your project
- Be conservative in referring to other projects' documentation
- Try not to fragment your documentation

Examples Examples Examples!

- Use lots of examples
- But make sure readers can easily reproduce them

Screenshots

- Screenshots should clarify, not distract!
- Leave out irrelevant stuff
- Use consistent resolution and theme
- Use consistent language settings
- Set up a VM with default OS/browser settings

Reference guide

- Provide a reference guide
- Glossary, concepts, fundamentals, syntax
- Complements “howto“ style guides
- Refer to/from tutorials and examples

Be specific about versions

- Always mention what version this documentation applies to
- Never refer to a version as "latest" or "current"
- Screenshots should reflect the version the documentation applies to

Test the documentation

- Try it out yourself, you might be surprised!
- Even better, have others test it
- Do it frequently, at least before each release

Watch out for outdated documentation

- Take outdated docs offline until they are updated
- Or update them right away!
- Leaving outdated docs online will bounce back to you

About documentation tools

- The tools you use should not be an excuse not to write documentation
- Wikis are great for initial ideas and user contributions, but they should be maintained
- Ultimately you will want a CMS

The devil is in the details

- Consistency
  - Terminology
  - Screenshots
- Completeness
  - If your give code examples, make sure readers know what to do with them
  - Test for dead links

Conclusion

- Producing decent documentation is not rocket science
- It just takes a lot of time
- But it's worth the effort
  - Happy users -> Adoption
  - User involvement
  - Healthy community