Going Modular: Teaching Old Docs New Tricks

DevConf.CZ 2019

Robert Krátký   @rkratky
Principal Technical Writer, Red Hat
1. Terminology, motivation
   a. What’s “old” & “new tricks”?
   b. Why bother?

2. What’s the problem?
   a. Too much docs
   b. Hard to find and navigate

3. What’s our solution?
   a. Document only what users really need
   b. Modular writing
TERMINOLOGY

- **Old docs** = big guides/books
  - comprehensive
  - feature-based

- **New tricks** = lean, concise articles (units)
  - targeted
  - (ideally) modular
WHY

Landscape is changing:

- **Software packaging**
  - Monolithic distributions ➔ containers, flatpak, ...

- **Software delivery**
  - Release cycles ➔ continuous deployment
WHAT’S THE PROBLEM (with old docs)?

Too much content:

- Difficult to maintain
- Lots of duplication

Hard to find and navigate:

- Which guide explains what?
- Need to piece info from many places

Google it instead & go to SO
WHAT’S OUR SOLUTION?

Document only what users really need:

- Identify and validate **user stories**
- Cut down on the amount of content

Efficient use of resources

Modular writing:

- Templates make writing easier
- Modular pieces easier to chew off

Save time & bring contributors on board
### FEATURE-BASED VS USER STORY-BASED

**FEATURES**
Describe all available features; comprehensive

1. Using Onions
   1.1. Peeling Onions
   1.2. Cutting Onions
   1.3. Frying Onions
2. Using Eggs
   2.1. Breaking Eggs
   2.2. Mixing Eggs with Milk
   2.3. Frying Eggs

**USER STORIES***
Explain how to achieve specific goals; selective

1. Making an Onion Omelette
   1.1. Setting up Workplace
   1.2. Preparing Ingredients
   1.3. Selecting Spices
   1.4. Combining Ingredients
   1.5. Frying Omelette Mixture
   1.6. Serving the Omelette
   1.7. Other Onion Recipes

* As an amateur cook, I want to make an onion omelette, so that I can impress my friends.
ASSEMBLIES & MODULES

ASSEMBLY

Introduction

Concept module

Procedure module #1

Procedure module #2

Additional resources

ASSEMBLY

(documented user story)

MODULES

(reusable units of content)
ASSEMBLIES & MODULES

ASSEMBLY

Documentation realization of a user story
Title
action-oriented, specific, e.g. “Making an Onion Omelette”.

Purpose
What is going to be accomplished.

Prerequisites
Actions that need to be taken or conditions that need to be satisfied.
Title
What is explained, e.g.
“Understanding the Importance of Omelettes in French Cuisine”.

Concept
Concise, to the point.

Additional resources
Where to go for further details or background info.
**Title**
“Frying Omelette Mixture”.

(Purpose)
(Prerequisites)

Procedure
Commands, i.e.
1. Select ...
2. Open ...
3. Copy ...
4. ...

Additional resources
ASSEMBLIES & MODULES

ASSEMBLY

Introduction

- Concept module

- Procedure module #1

- Procedure module #2

- Additional resources

Links & references to:

- Related assemblies
- Individual modules
  - Concepts
  - References
  - Related procedures
- Websites
- Manual pages
- Other onion recipes
- ...

https://github.com/redhat-documentation/modular-docs
MODULAR PRESENTATION

- Filtering based on metadata
  - Entire body of modules and assemblies considered
- Presentation of books, user stories, modules
  - Preserving book experience
  - Adding more granular browsing possibilities
- Dynamic sorting and assembling
  - Metadata-based generation of doc sets
  - Smart arrangement
- Customized user experience
  - Preselected content based on product use
  - Recommendations
OLD
Long, feature-based guides, going after completeness

Lots of content, hard to maintain, learning curve for contributors too steep

Cumbersome navigation, static content, no customization

NEW
Adapting into modular, action-oriented units, based on user stories

Cut down on the amount of content, small chunks = low-hanging fruit for new people

Hierarchical, metadata-based navigation
DISCUSSION

- Do you use something similar?
- What are the major risks and/or caveats of this approach?
  - No clear content strategy
    - tons of assemblies
    - no real order
      → mind mapping?
  - Initial overhead
    - converting existing docs
    - maintenance of legacy docs
      → work progressively; while fixing old docs, adapt into modular
THANK YOU

✉️ rkratky@redhat.com
🐦 @rkratky