



KubeCon



CloudNativeCon

Europe 2021

Virtual



Forward Together »



KubeCon



CloudNativeCon

Europe 2021

Writing for Developers: take your project docs to the next level

Celeste Horgan, Sr. Technical Writer – CNCF

Virtual



@celeste_horgan | celeste.works

Fun fact



Virtual

About **50%* of responses** to “how to choose a development framework” questions on StackOverflow mention **documentation**.

* This is a rough estimate done by me

A call to arms

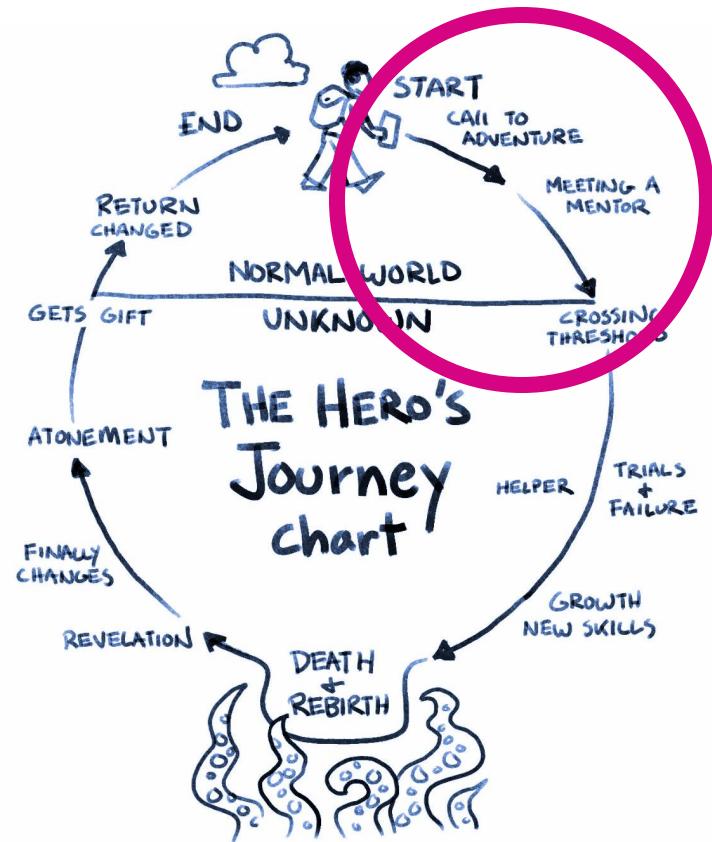


Virtual

CNCF is a community of **open source** projects. As far an open source project itself goes, there is no dedicated **support team**, no **sales**, and no **dedicated writers!**

We are what our partner organizations choose to staff, and what the community **chooses to contribute to**. And if we don't choose docs, they don't get done.

The hero's journey



Little Red Riding Hood



KubeCon



CloudNativeCon

Europe 2021

Virtual



Your user has a **clear goal**.
Your first task is to understand it.

@celeste_horgan | celeste.works

What is your user's goal?



Virtual

Learning about user goals

Talk to your users

Keep track of questions asked repeatedly

Outline the docs as a part of the epic

How new contributors can help

Submit GitHub issues and PRs

Make suggestions to the project team

Cherish your beginner's mindset

User goals and how to meet them



KubeCon



CloudNativeCon

Europe 2021

Virtual

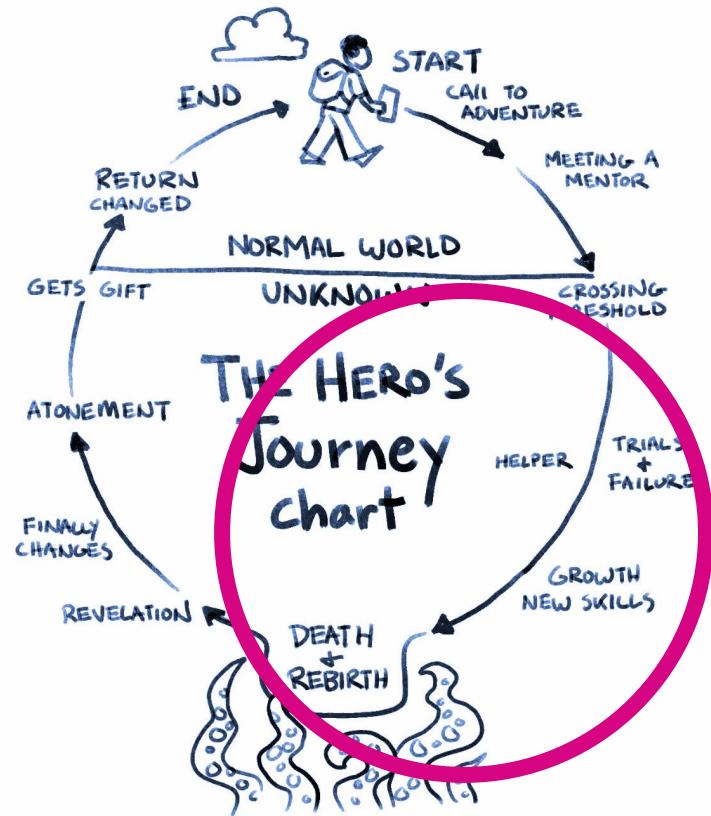
Common user goals

- Evaluating new tech stacks
- Learning, passing certification exams
- Supporting an existing system
- Finding info to solve a problem

How to meet them

- Concept docs, Quickstarts
- (API) References
- Tasks, tutorials and cookbooks
- Searchability & information architecture

The hero's journey



Hercules



KubeCon



CloudNativeCon

Virtual

Europe 2021



Your user needs **new skills**
to accomplish their goals.

How to explain anything



KubeCon



CloudNativeCon

Europe 2021

Virtual

Concept

Explain at a high level



Why do they care?

Tasks

How do they do it?

Where are the pitfalls?



Reference

The long tail explanation

Lists of things

Concepts

Concept topics explain a feature at a high level.

- Often use “_____ is a _____ which _____. ” sentence construction
- Briefly describes or links to other concepts related to the feature at hand
- Describes why a user might care about a feature, or common use cases.

The screenshot shows a web browser window with the title "The Vitess Docs | Cell". The URL in the address bar is <https://vitess.io/docs/concepts/cell>. The page content starts with the word "Cell" in large font. Below it, a definition is provided: "Data center, availability zone or group of computing resources". There are two buttons at the bottom left: "Edit" and "Documentation". At the bottom right, there is a link "Execution Plans >>".

A *cell* is a group of servers and network infrastructure collocated in an area, and isolated from failures in other cells. It is typically either a full data center or a subset of a data center, sometimes called a *zone* or *availability zone*. Vitess gracefully handles cell-level failures, such as when a cell is cut off the network.

Each cell in a Vitess implementation has a **local topology service**, which is hosted in that cell. The topology service contains most of the information about the Vitess tablets in its cell. This enables a cell to be taken down and rebuilt as a unit.

Tasks



Virtual

Tasks describe what a user can do with a feature.

- Describe each step in sequence
- Install and quickstarts are task topics
- Simple tasks often map to CRUD (create, read, update, delete) functionality.
- Complex tasks which weave through many concepts/features are often written as tutorials
- Example code focused task topics like cookbooks or example repos work too, but should never be treated as standalone!

A screenshot of a web browser window titled "The Vitess Docs | Local Install". The URL in the address bar is https://vitess.io/docs/get-started/local-install/. The page content is titled "Local Install via Homebrew". It includes instructions for using Vitess on a macOS machine for testing purposes, a "Documentation" button, and navigation links for "Local Install" and "Vitess Operator for Kuber...".

Local Install via Homebrew

Instructions for using Vitess on your macOS machine for testing purposes

Edit Documentation ▾

<< Local Install Vitess Operator for Kuber... >>

This guide covers installing Vitess locally to macOS for testing purposes, from pre-compiled binaries. We will launch multiple copies of `mysqld`, so it is recommended to have greater than 4GB RAM, as well as 20GB of available disk space.

A [Homebrew](#) package manager is also available, which requires no dependencies on your local host.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/vitessio/vitess/develop/debian/install.sh)"
```

References



Virtual

References provide lists of all options for a feature

- Often autogenerated from code comments or specs
- API references describe all endpoints and accepted data
- Command line interface flags, Javadoc-style documentation
- Completeness and accuracy is important here!
- Complex APIs benefit from request/response format code samples

A screenshot of a web browser window displaying the Kubernetes ResourceQuota API reference page. The URL in the address bar is https://kubernetes.io/docs/reference/api/core/v1/ResourceQuota. The page has a dark header with the Kubernetes logo and a search bar. The main content area is titled "ResourceQuota" and contains YAML code for the API object. Below the code, there is a detailed description of the ResourceQuota object, its fields, and examples of how to use it. A sidebar on the right shows navigation links and a search bar.

```
apiVersion: v1  
import "k8s.io/api/core/v1"
```

ResourceQuota

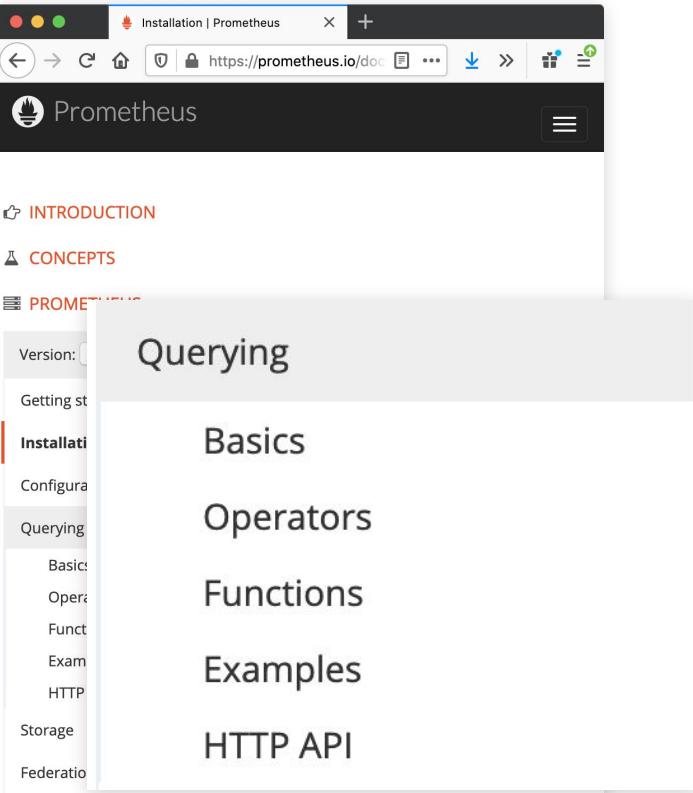
ResourceQuota sets aggregate quota restrictions enforced per namespace

- **apiVersion:** v1
- **kind:** ResourceQuota
- **metadata** ([ObjectMeta](#))

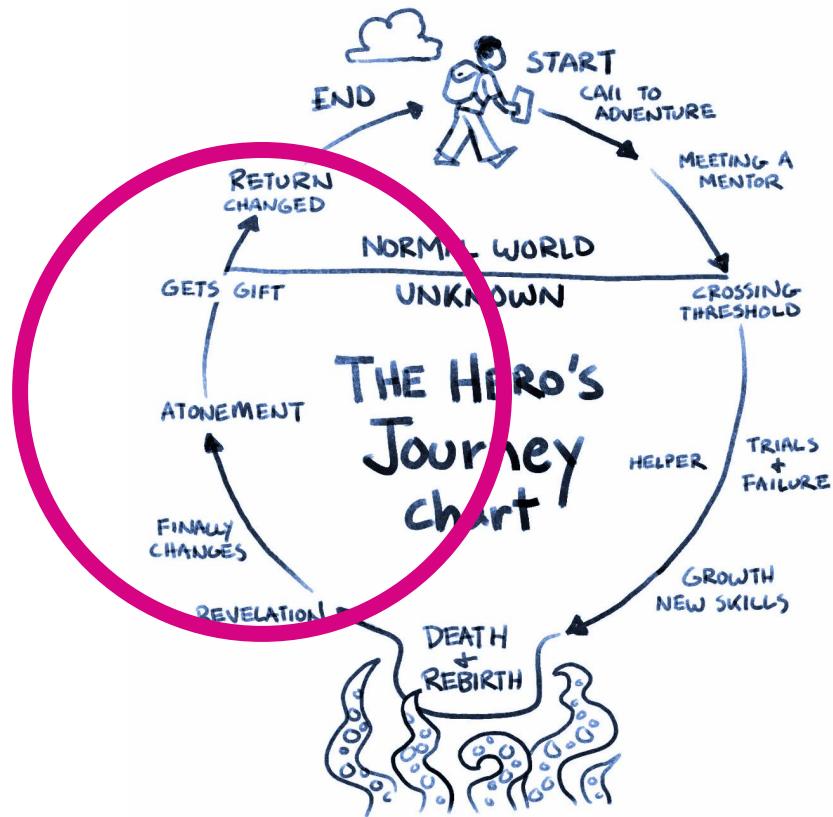
Standard object's metadata. More details in [ObjectMeta](#).
[/community/contributors/conventions.md#metadata](#)

Putting it all together

Great docs almost always have all 3 types (concept, reference, task) present in some format.



The hero's journey



The hero returns, changed



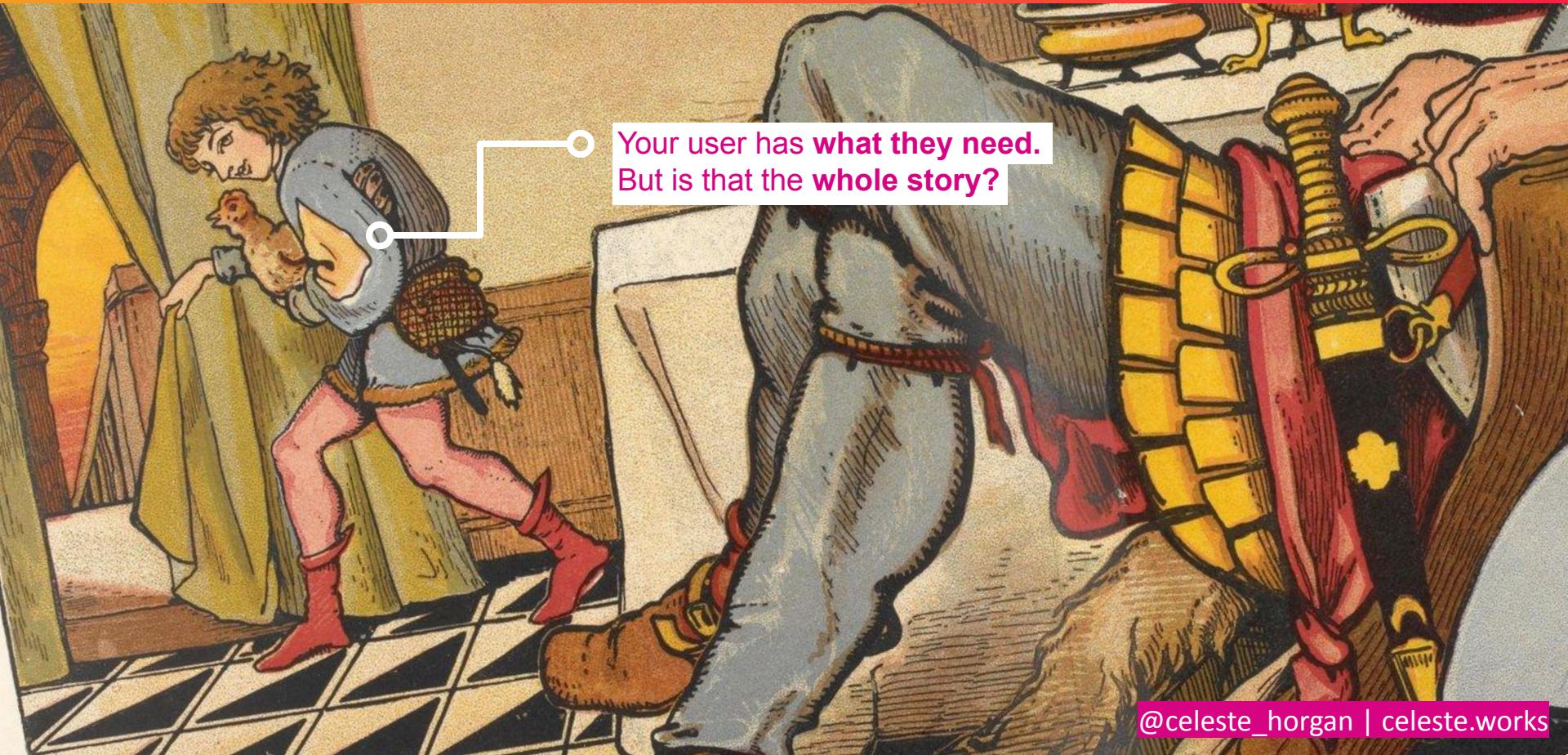
KubeCon



CloudNativeCon

Europe 2021

Virtual



User flow



Virtual

Great documentation shows its users where to go next

- Organize the entire doc set along a user flow: what can you expect people to do first? Second? Third? Last?
- Within a document, organize information from more general (concept) to more specific (task, then reference)
- Well-curated “learn more” links never go out of style

A screenshot of a web browser displaying the Kubernetes Components documentation page. The page has a dark header with a blue gear icon and the text "Kubernetes Components | Kube X". The main content area has a white background. At the top, there is a navigation bar with icons for back, forward, search, and other browser functions. Below the navigation, there is a large blue gear icon and a three-line menu icon. The main content starts with a section titled "What's next" containing a bulleted list of links: "Learn about Nodes", "Learn about Controllers", "Learn about kube-scheduler", and "Read etcd's official documentation". Below this is a section titled "Feedback" with the question "Was this page helpful?". There are two blue circular buttons labeled "Yes" and "No". At the bottom of the page, there is a footer note: "Last modified January 03, 2021 at 10:22 AM PST: Modified description (976c260ef)".

Docs are a team sport

New contributors and end users are key to great documentation: everyone needs help with their blind spots!



Virtual



@celeste_horgan | celeste.works

Thank you!



Virtual

Where to go...

- For CNCF projects looking for help with docs: Join us at CNCF techdocs office hours monthly!
- For new writers looking to help out: Any project's community meetings, or Kubernetes SIG Docs
- To find me: I tweet at [@celeste_horgan](https://twitter.com/celeste_horgan), or find me on CNCF and Kubernetes Slack!