# Welcome to Kubernetes

or:

## I’m here. Where do I start?

*Guinevere Saenger*

*Software Developer*

*Samsung Cloud Native Computing Team*

KubeCon

North America 2017

SAMSUNG SDS

SAMSUNG

# Fun Facts About Me

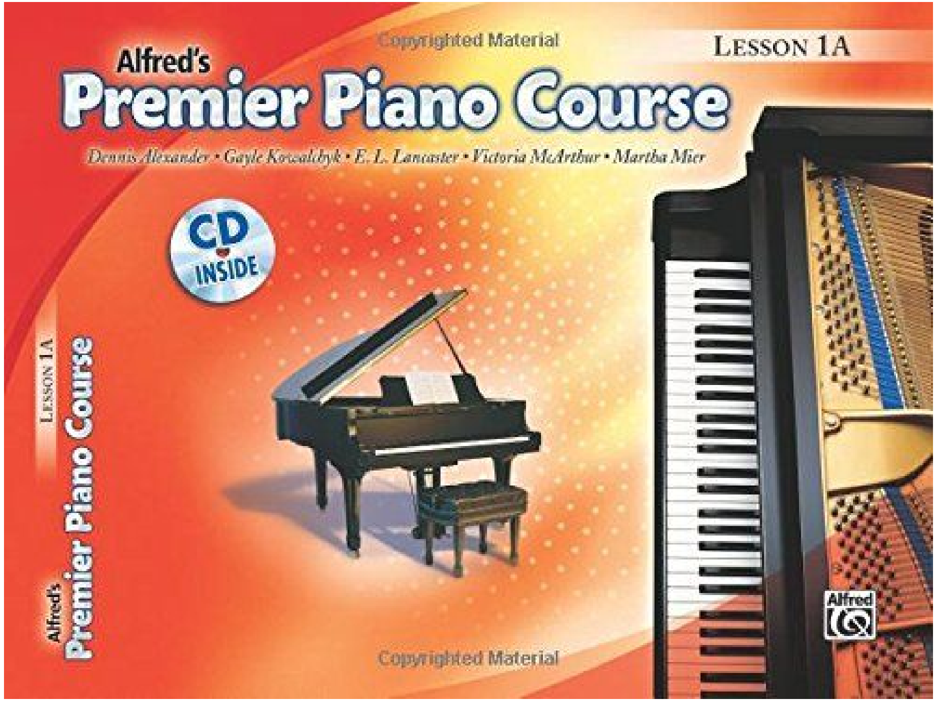
I took my first programming class a little less than 2 years ago.

This time, last year, I had no idea what Kubernetes was.

Thanks to Ada Developers Academy and the Cloud Native team at Samsung, I am here today.

# This is what I used to do for a living

# What I *actually* did for a living

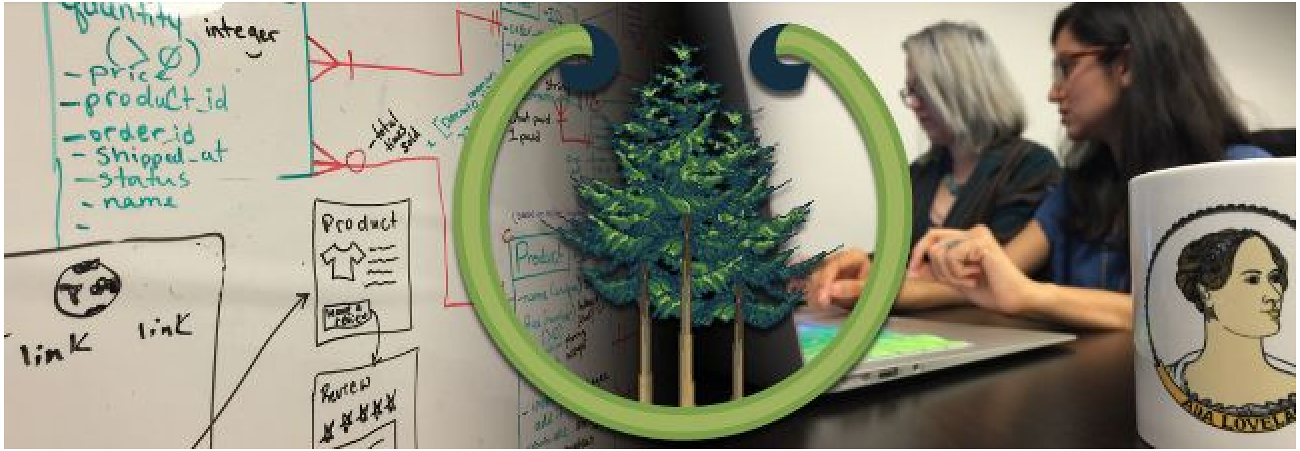

Copyrighted Material
Lesson 1A
Alfred's
Premier Piano Course
Dennis Alexander • Gayle Kowalchyk • E. L. Lancaster • Victoria McArthur • Martha Mier
CD INSIDE
Lesson 1A
Alfred's Premier Piano Course
Alfred
Copyrighted Material

You can do this.

# Ada Developers Academy

## Take a year to become a software engineer

(*Diversify the tech sector while you’re at it*)



SAMSUNG SDS SAMSUNG

# The Internship

Half of the time at Ada is spent in a tech internship

My internship turned out to be at Samsung SDS

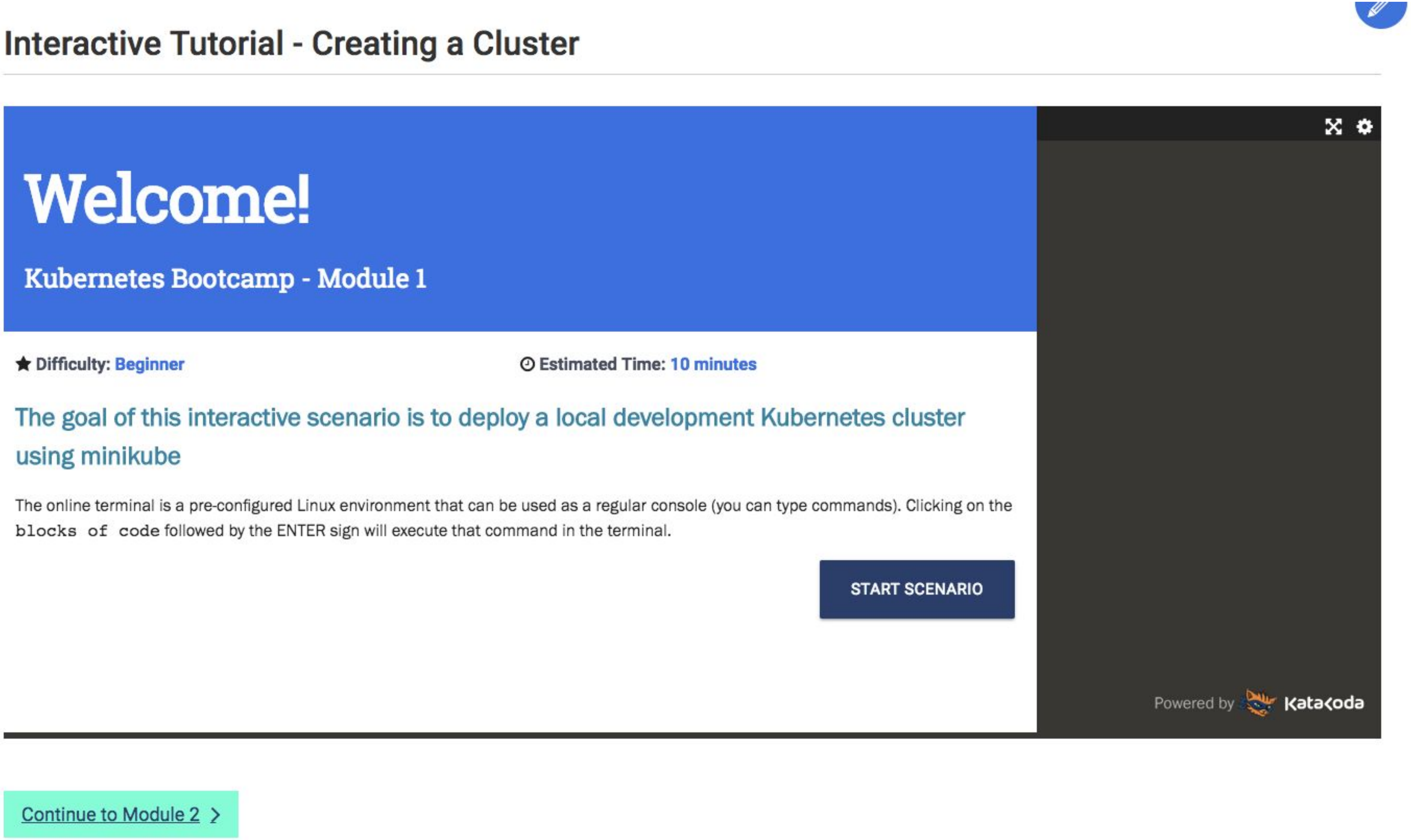
I just happened to slip into a situation where I was working with Kubernetes. Fortunately, Kubernetes is really cool!

# I Got The Talk, Now What.

# First Step: Practical Exposure

# Kubernetes Basics Tutorial

https://kubernetes.io/docs/tutorials/kubernetes-basics/



## Interactive Tutorial - Creating a Cluster

# Welcome!

### Kubernetes Bootcamp - Module 1

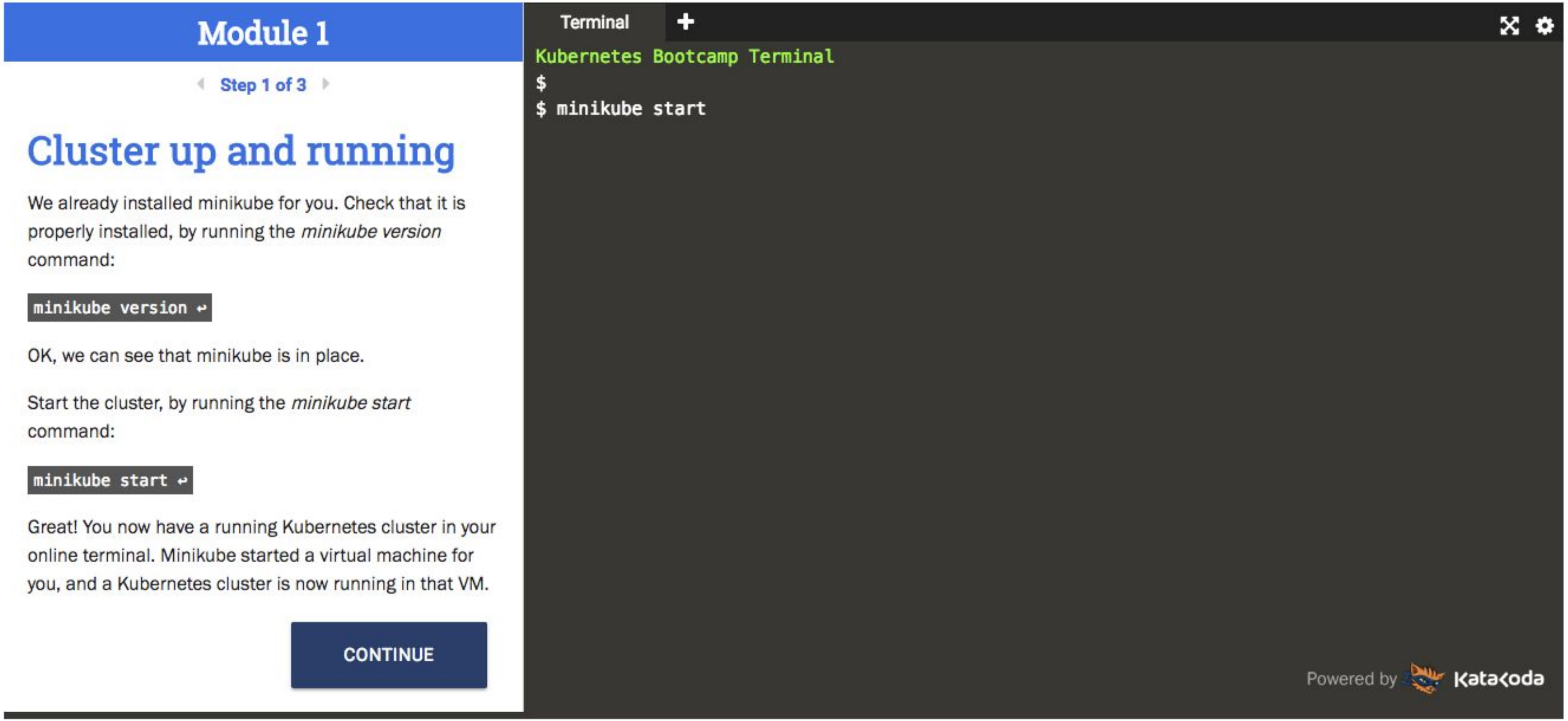
★ Difficulty: Beginner

Estimated Time: 10 minutes

### The goal of this interactive scenario is to deploy a local development Kubernetes cluster using minikube

The online terminal is a pre-configured Linux environment that can be used as a regular console (you can type commands). Clicking on the `blocks of code` followed by the ENTER sign will execute that command in the terminal.

# Interactive Tutorial - Creating a Cluster



## Cluster up and running

We already installed minikube for you. Check that it is properly installed, by running the *minikube version* command:

```
minikube version ↩
```

OK, we can see that minikube is in place.

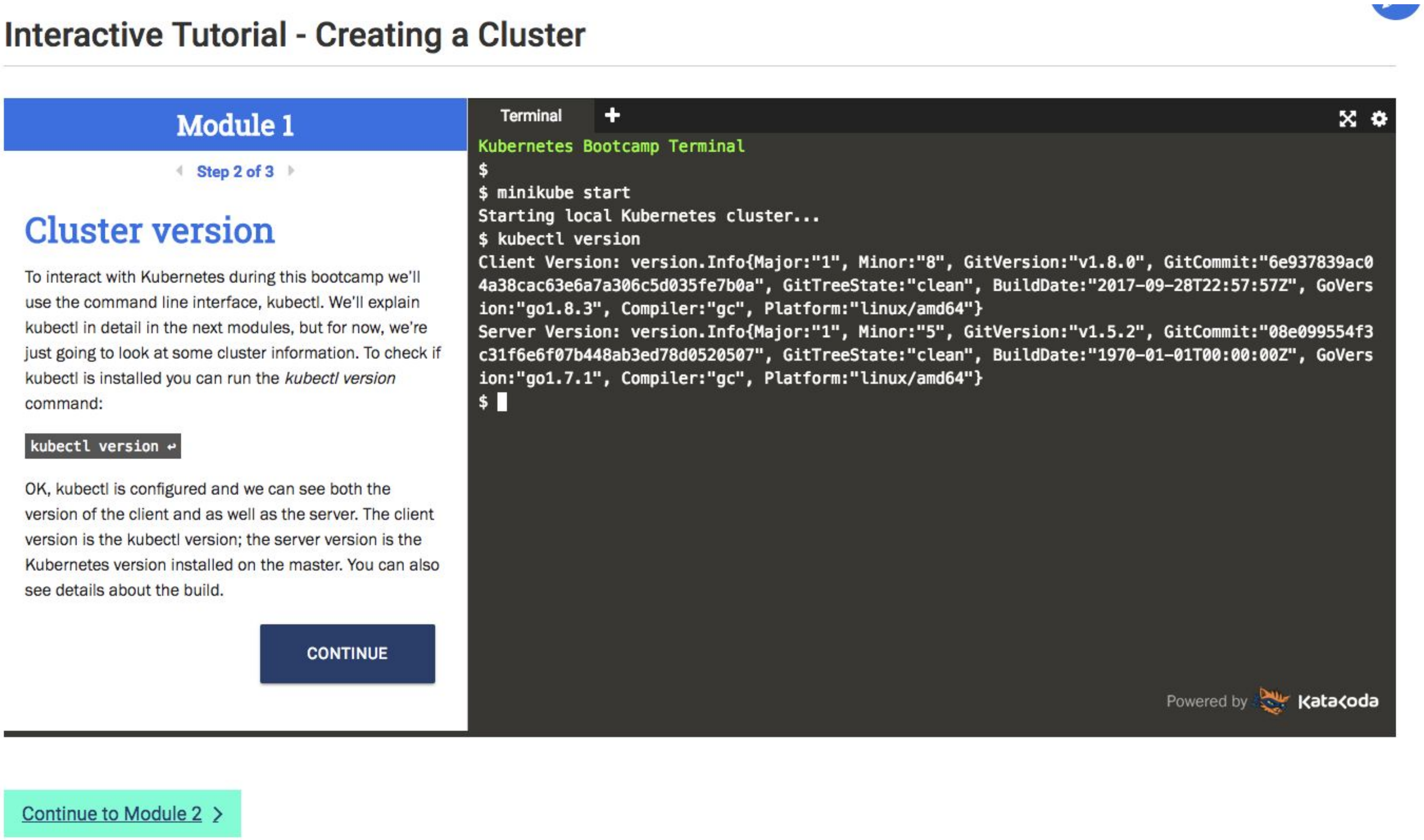
Start the cluster, by running the *minikube start* command:

```
minikube start ↩
```

Great! You now have a running Kubernetes cluster in your online terminal. Minikube started a virtual machine for you, and a Kubernetes cluster is now running in that VM.

```
Kubernetes Bootcamp Terminal
$
$ minikube start
```

Continue to Module 2 >



# Interactive Tutorial - Creating a Cluster

## Module 1

Step 2 of 3

## Cluster version

To interact with Kubernetes during this bootcamp we'll use the command line interface, kubectl. We'll explain kubectl in detail in the next modules, but for now, we're just going to look at some cluster information. To check if kubectl is installed you can run the *kubectl version* command:

```
kubectl version ↵
```

OK, kubectl is configured and we can see both the version of the client and as well as the server. The client version is the kubectl version; the server version is the Kubernetes version installed on the master. You can also see details about the build.

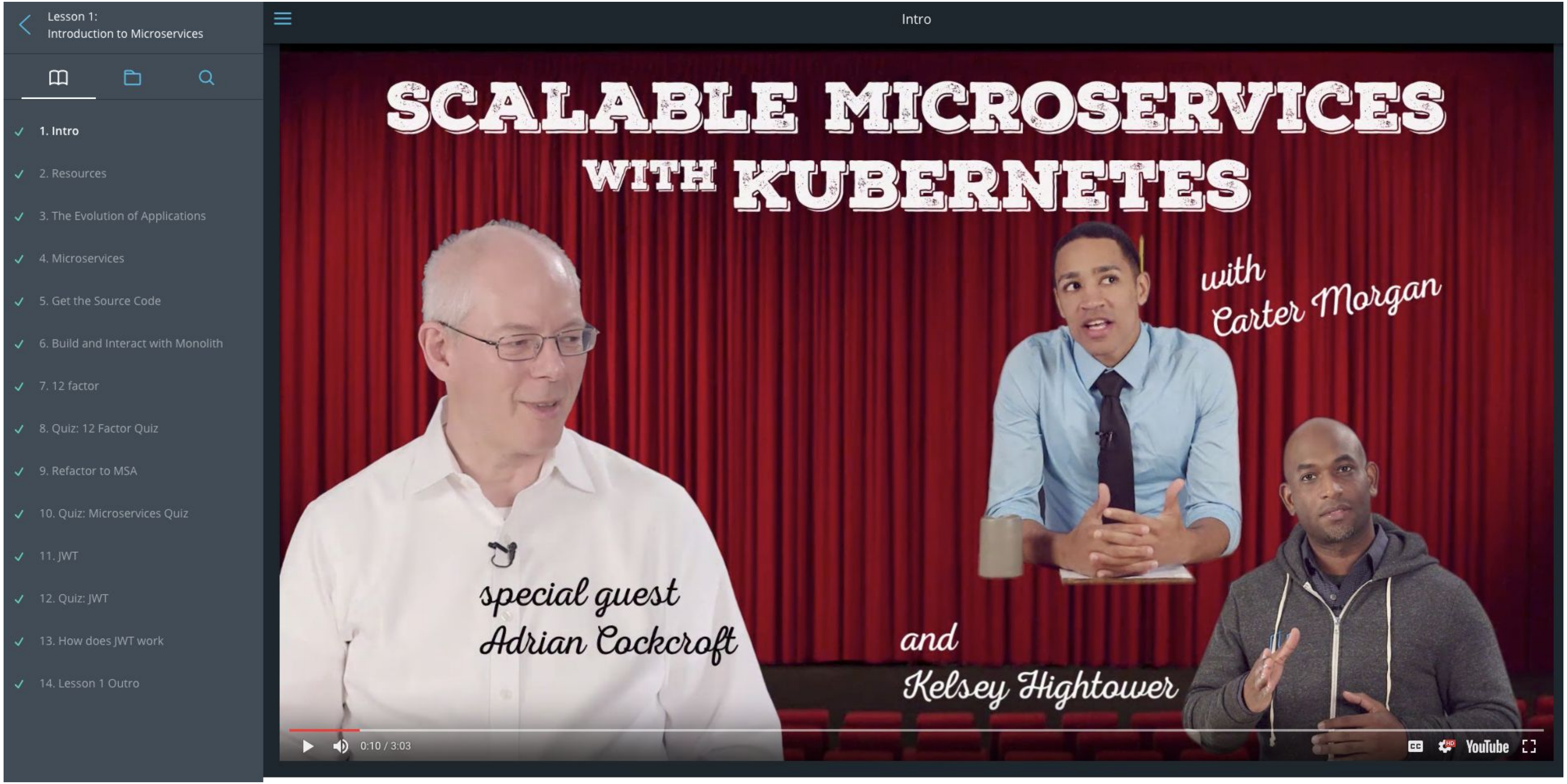
```
Kubernetes Bootcamp Terminal
$
$ minikube start
Starting local Kubernetes cluster...
$ kubectl version
Client Version: version.Info{Major:"1", Minor:"8", GitVersion:"v1.8.0", GitCommit:"6e937839ac0
4a38cac63e6a7a306c5d035fe7b0a", GitTreeState:"clean", BuildDate:"2017-09-28T22:57:57Z", GoVers
ion:"go1.8.3", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"5", GitVersion:"v1.5.2", GitCommit:"08e099554f3
c31f6e6f07b448ab3ed78d0520507", GitTreeState:"clean", BuildDate:"1970-01-01T00:00:00Z", GoVers
ion:"go1.7.1", Compiler:"gc", Platform:"linux/amd64"}
$
```

# Udacity - Scalable Microservices With Kubernetes

https://www.udacity.com/course/scalable-microservices-with-kubernetes--ud615

(you’ll need to sign up for udacity, but it’s free)

# Video mini-lectures...

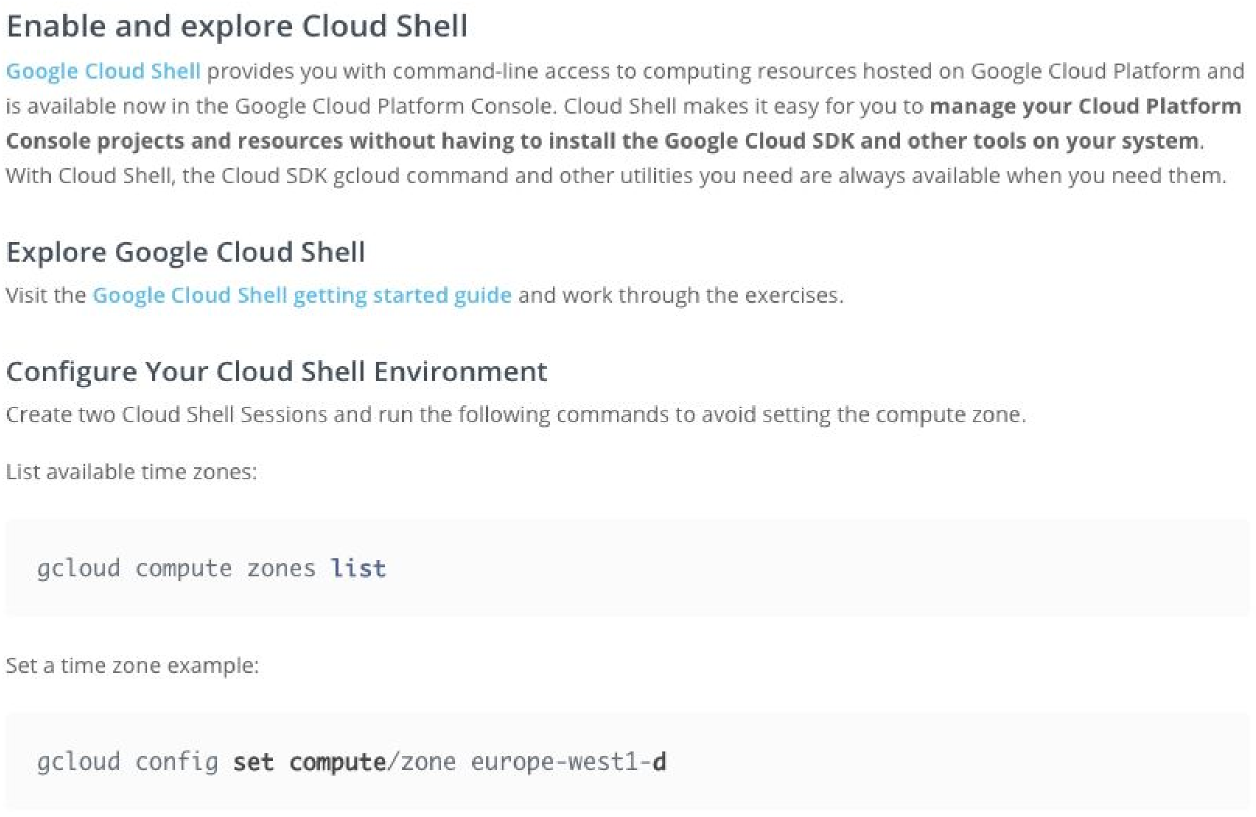


Lesson 1:
Introduction to Microservices

Intro

- ✓ 1. Intro
- ✓ 2. Resources
- ✓ 3. The Evolution of Applications
- ✓ 4. Microservices
- ✓ 5. Get the Source Code
- ✓ 6. Build and Interact with Monolith
- ✓ 7. 12 factor
- ✓ 8. Quiz: 12 Factor Quiz
- ✓ 9. Refactor to MSA
- ✓ 10. Quiz: Microservices Quiz
- ✓ 11. JWT
- ✓ 12. Quiz: JWT
- ✓ 13. How does JWT work
- ✓ 14. Lesson 1 Outro

# Step-by-step instruction...



## Enable and explore Cloud Shell

Google Cloud Shell provides you with command-line access to computing resources hosted on Google Cloud Platform and is available now in the Google Cloud Platform Console. Cloud Shell makes it easy for you to **manage your Cloud Platform Console projects and resources without having to install the Google Cloud SDK and other tools on your system**. With Cloud Shell, the Cloud SDK gcloud command and other utilities you need are always available when you need them.

## Explore Google Cloud Shell

Visit the Google Cloud Shell getting started guide and work through the exercises.

## Configure Your Cloud Shell Environment

Create two Cloud Shell Sessions and run the following commands to avoid setting the compute zone.

List available time zones:

```
gcloud compute zones list
```

Set a time zone example:

```
gcloud config set compute/zone europe-west1-d
```

# Explore the docs!

https://kubernetes.io/docs/home/

# The Kubernetes Docs

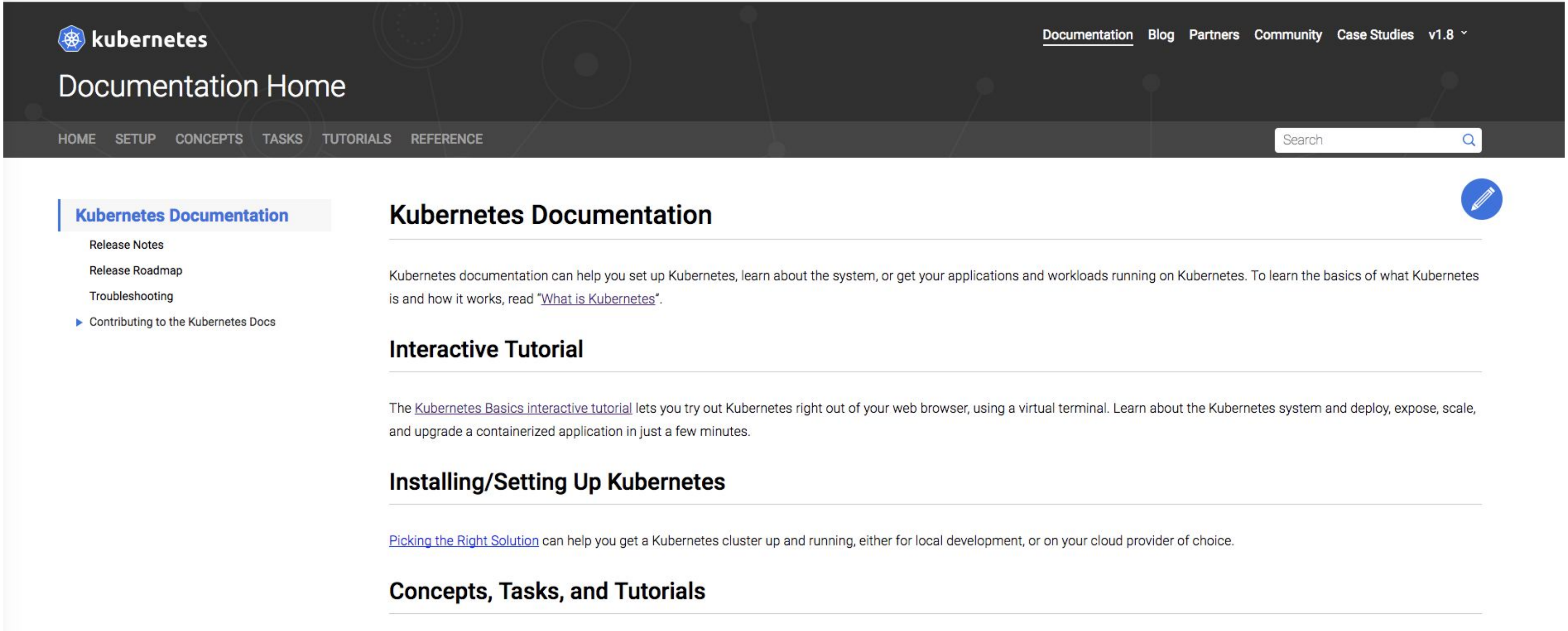
The K8s docs have two important sections:

- Concepts

  Sample configurations for components, explanations, mini demos

- Reference

  Technical reference for all the parts of all the components



# Documentation Home

## Kubernetes Documentation

Kubernetes documentation can help you set up Kubernetes, learn about the system, or get your applications and workloads running on Kubernetes. To learn the basics of what Kubernetes is and how it works, read "What is Kubernetes".

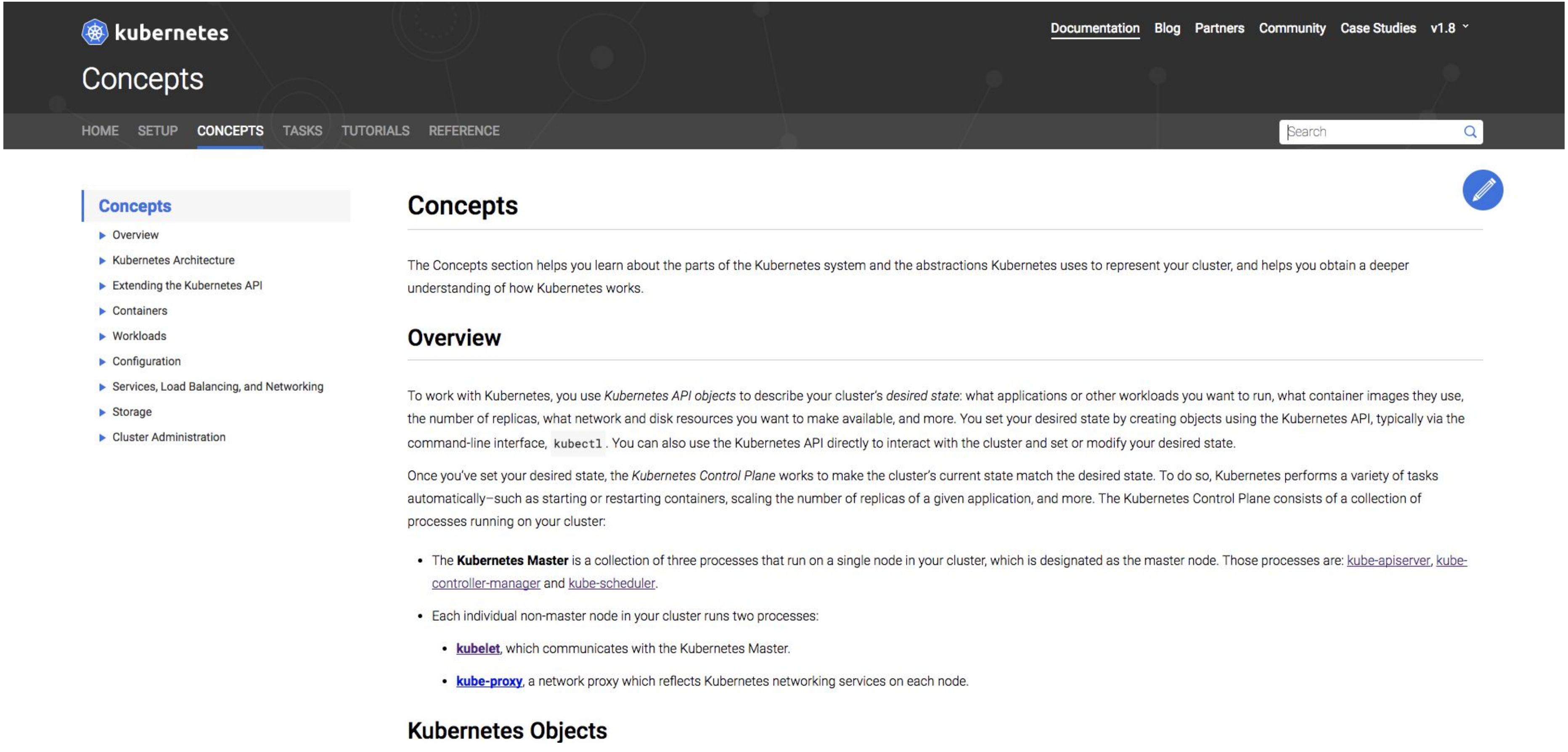
## Interactive Tutorial

The Kubernetes Basics interactive tutorial lets you try out Kubernetes right out of your web browser, using a virtual terminal. Learn about the Kubernetes system and deploy, expose, scale, and upgrade a containerized application in just a few minutes.

## Installing/Setting Up Kubernetes

Picking the Right Solution can help you get a Kubernetes cluster up and running, either for local development, or on your cloud provider of choice.

## Concepts, Tasks, and Tutorials



# Concepts

The Concepts section helps you learn about the parts of the Kubernetes system and the abstractions Kubernetes uses to represent your cluster, and helps you obtain a deeper understanding of how Kubernetes works.

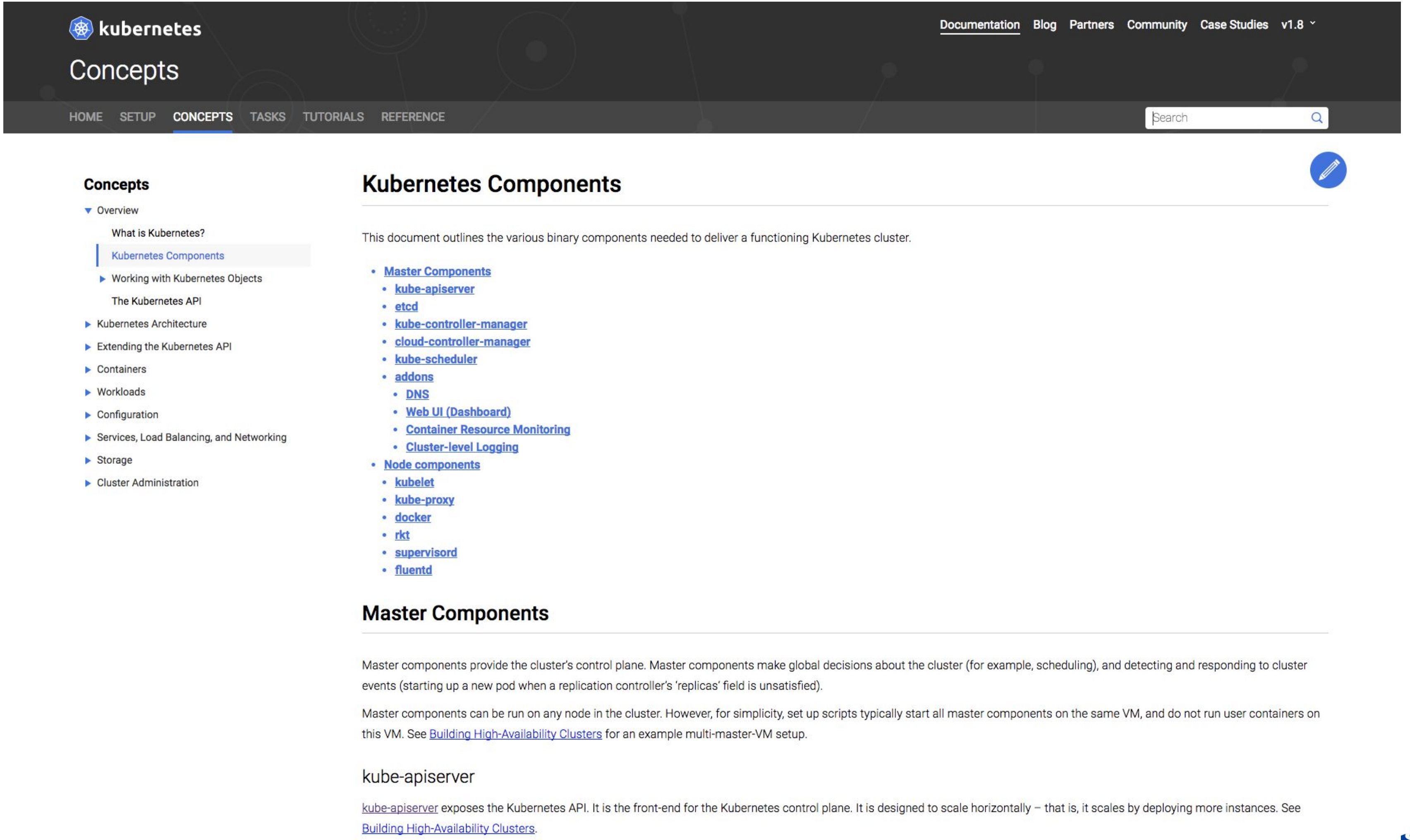
## Overview

To work with Kubernetes, you use *Kubernetes API objects* to describe your cluster's *desired state*: what applications or other workloads you want to run, what container images they use, the number of replicas, what network and disk resources you want to make available, and more. You set your desired state by creating objects using the Kubernetes API, typically via the command-line interface, `kubectl`. You can also use the Kubernetes API directly to interact with the cluster and set or modify your desired state.

Once you've set your desired state, the *Kubernetes Control Plane* works to make the cluster's current state match the desired state. To do so, Kubernetes performs a variety of tasks automatically–such as starting or restarting containers, scaling the number of replicas of a given application, and more. The Kubernetes Control Plane consists of a collection of processes running on your cluster:

- The **Kubernetes Master** is a collection of three processes that run on a single node in your cluster, which is designated as the master node. Those processes are: kube-apiserver, kube-controller-manager and kube-scheduler.
- Each individual non-master node in your cluster runs two processes:
  - **kubelet**, which communicates with the Kubernetes Master.
  - **kube-proxy**, a network proxy which reflects Kubernetes networking services on each node.

## Kubernetes Objects



# Kubernetes Components

This document outlines the various binary components needed to deliver a functioning Kubernetes cluster.

- Master Components
  - kube-apiserver
  - etcd
  - kube-controller-manager
  - cloud-controller-manager
  - kube-scheduler
  - addons
    - DNS
    - Web UI (Dashboard)
    - Container Resource Monitoring
    - Cluster-level Logging
- Node components
  - kubelet
  - kube-proxy
  - docker
  - rkt
  - supervisord
  - fluentd

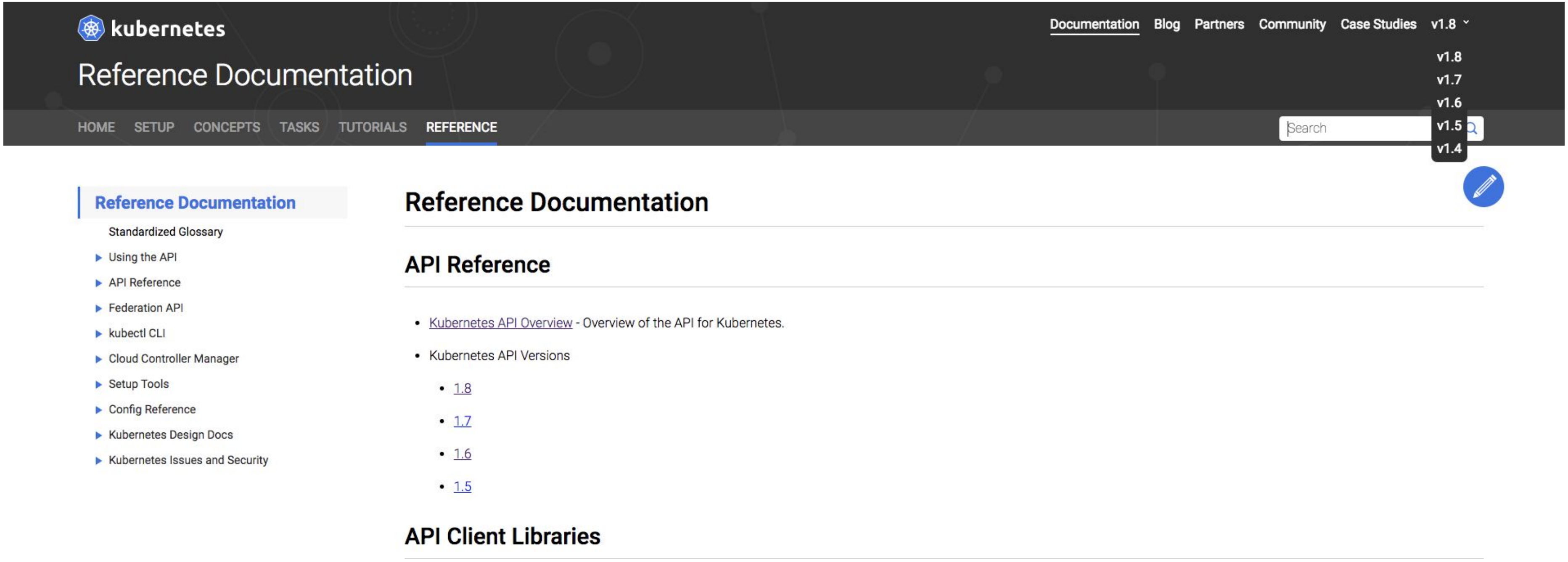
## Master Components

Master components provide the cluster's control plane. Master components make global decisions about the cluster (for example, scheduling), and detecting and responding to cluster events (starting up a new pod when a replication controller's 'replicas' field is unsatisfied).

Master components can be run on any node in the cluster. However, for simplicity, set up scripts typically start all master components on the same VM, and do not run user containers on this VM. See Building High-Availability Clusters for an example multi-master-VM setup.

### kube-apiserver

kube-apiserver exposes the Kubernetes API. It is the front-end for the Kubernetes control plane. It is designed to scale horizontally – that is, it scales by deploying more instances. See Building High-Availability Clusters.



# Reference Documentation

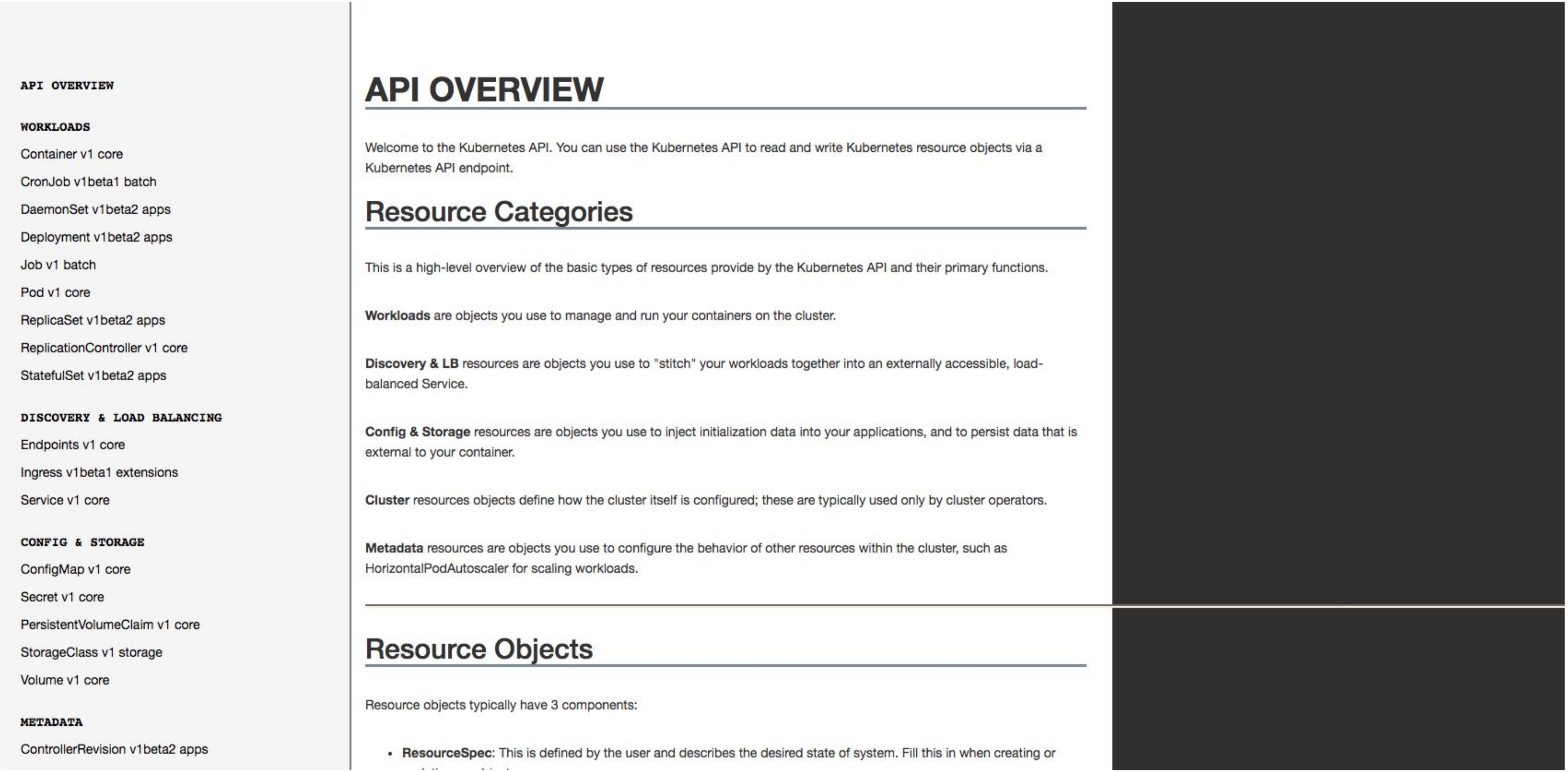
- Reference Documentation
  - Standardized Glossary
  - Using the API
  - API Reference
  - Federation API
  - kubectl CLI
  - Cloud Controller Manager
  - Setup Tools
  - Config Reference
  - Kubernetes Design Docs
  - Kubernetes Issues and Security

## Reference Documentation

## API Reference

- Kubernetes API Overview - Overview of the API for Kubernetes.
- Kubernetes API Versions
  - 1.8
  - 1.7
  - 1.6
  - 1.5

## API Client Libraries



# API OVERVIEW

Welcome to the Kubernetes API. You can use the Kubernetes API to read and write Kubernetes resource objects via a Kubernetes API endpoint.

## Resource Categories

This is a high-level overview of the basic types of resources provide by the Kubernetes API and their primary functions.

**Workloads** are objects you use to manage and run your containers on the cluster.

**Discovery & LB** resources are objects you use to "stitch" your workloads together into an externally accessible, load-balanced Service.

**Config & Storage** resources are objects you use to inject initialization data into your applications, and to persist data that is external to your container.

**Cluster** resources objects define how the cluster itself is configured; these are typically used only by cluster operators.

**Metadata** resources are objects you use to configure the behavior of other resources within the cluster, such as HorizontalPodAutoscaler for scaling workloads.

## Resource Objects

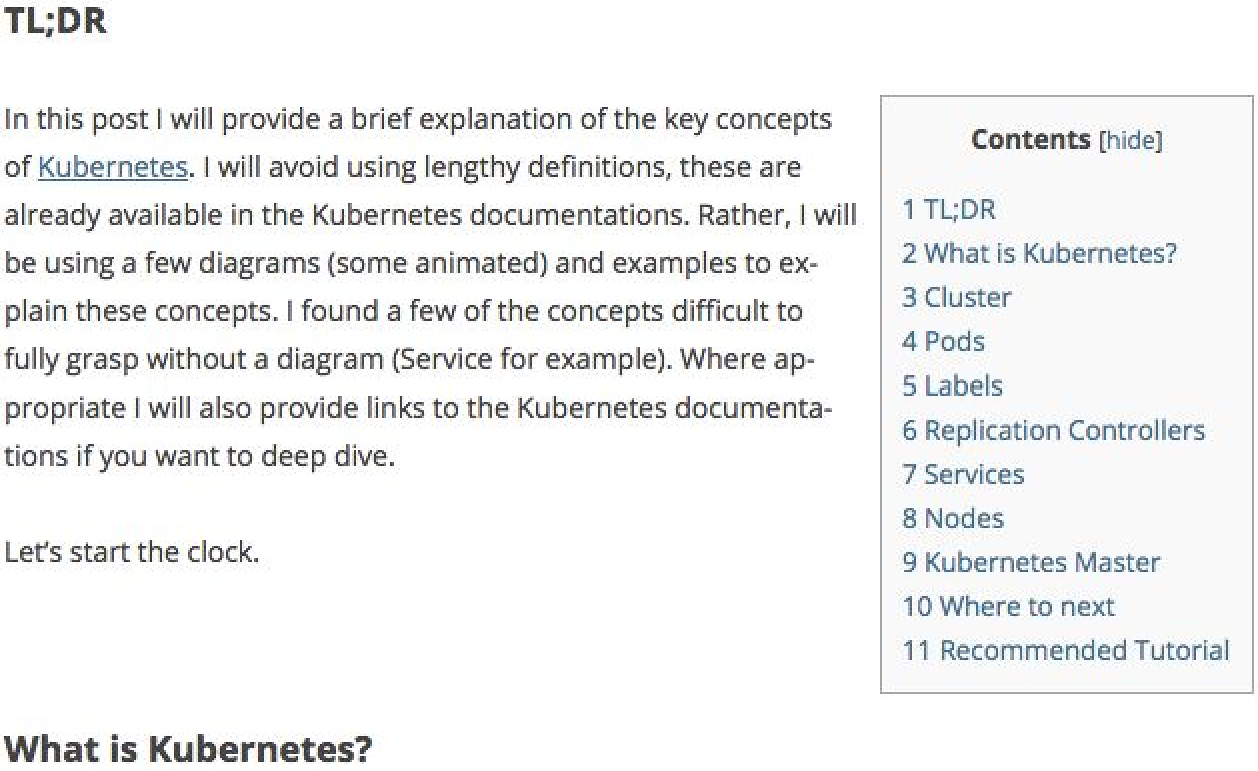
Resource objects typically have 3 components:

- **ResourceSpec**: This is defined by the user and describes the desired state of system. Fill this in when creating or

# Quick reference

# Kubernetes Concepts in ten minutes

http://omerio.com/2015/12/18/learn-the-kubernetes-key-concepts-in-10-minutes/



## TL;DR

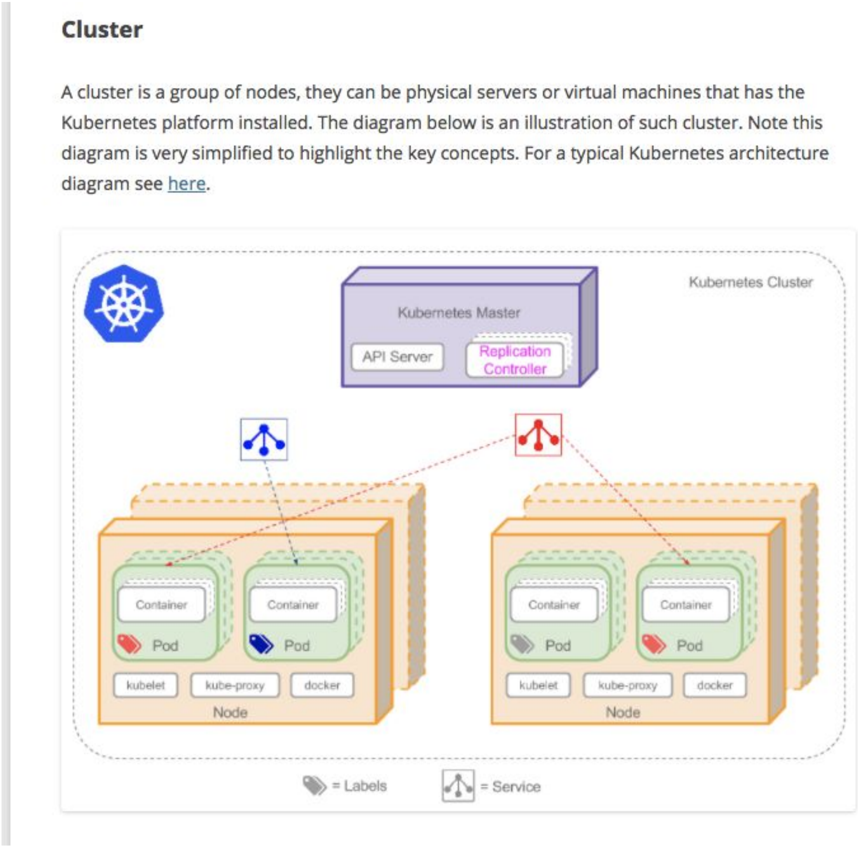
In this post I will provide a brief explanation of the key concepts of Kubernetes. I will avoid using lengthy definitions, these are already available in the Kubernetes documentations. Rather, I will be using a few diagrams (some animated) and examples to explain these concepts. I found a few of the concepts difficult to fully grasp without a diagram (Service for example). Where appropriate I will also provide links to the Kubernetes documentations if you want to deep dive.

Let's start the clock.

**Contents** [hide]

## What is Kubernetes?

# Comes with pretty pictures...

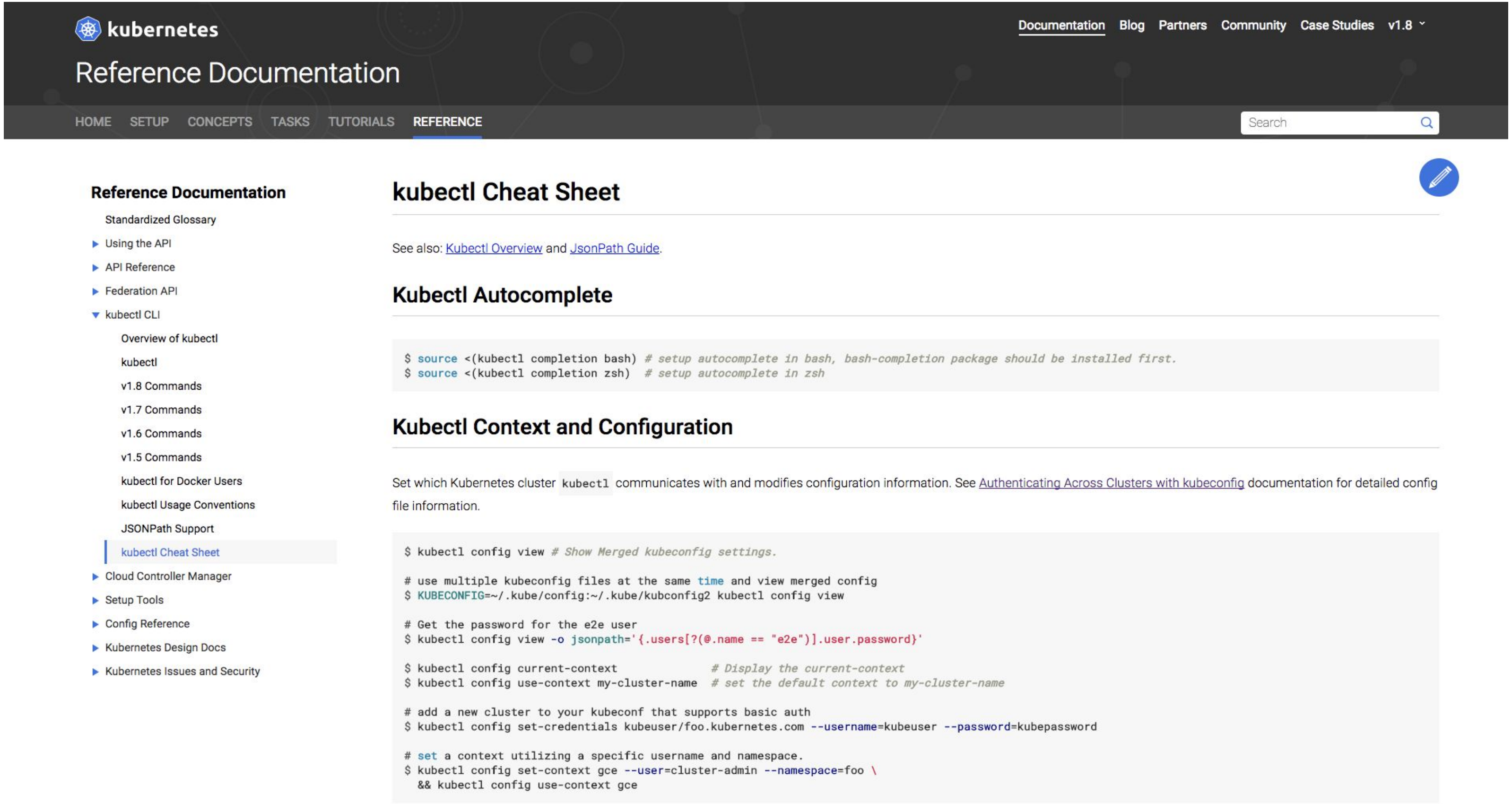


## Cluster

A cluster is a group of nodes, they can be physical servers or virtual machines that has the Kubernetes platform installed. The diagram below is an illustration of such cluster. Note this diagram is very simplified to highlight the key concepts. For a typical Kubernetes architecture diagram see here.

# Kubectl cheat sheet

https://kubernetes.io/docs/user-guide/kubectl-cheatsheet/



## kubectl Cheat Sheet

See also: Kubectl Overview and JsonPath Guide.

### Kubectl Autocomplete

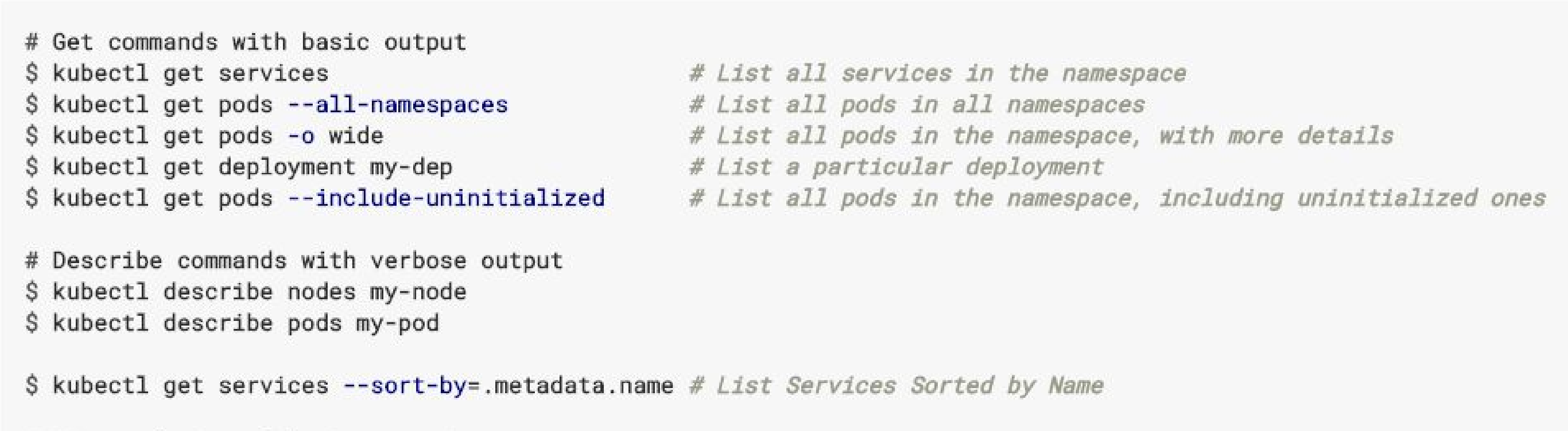
```
$ source <(kubectl completion bash) # setup autocomplete in bash, bash-completion package should be installed first.
$ source <(kubectl completion zsh)  # setup autocomplete in zsh
```

### Kubectl Context and Configuration

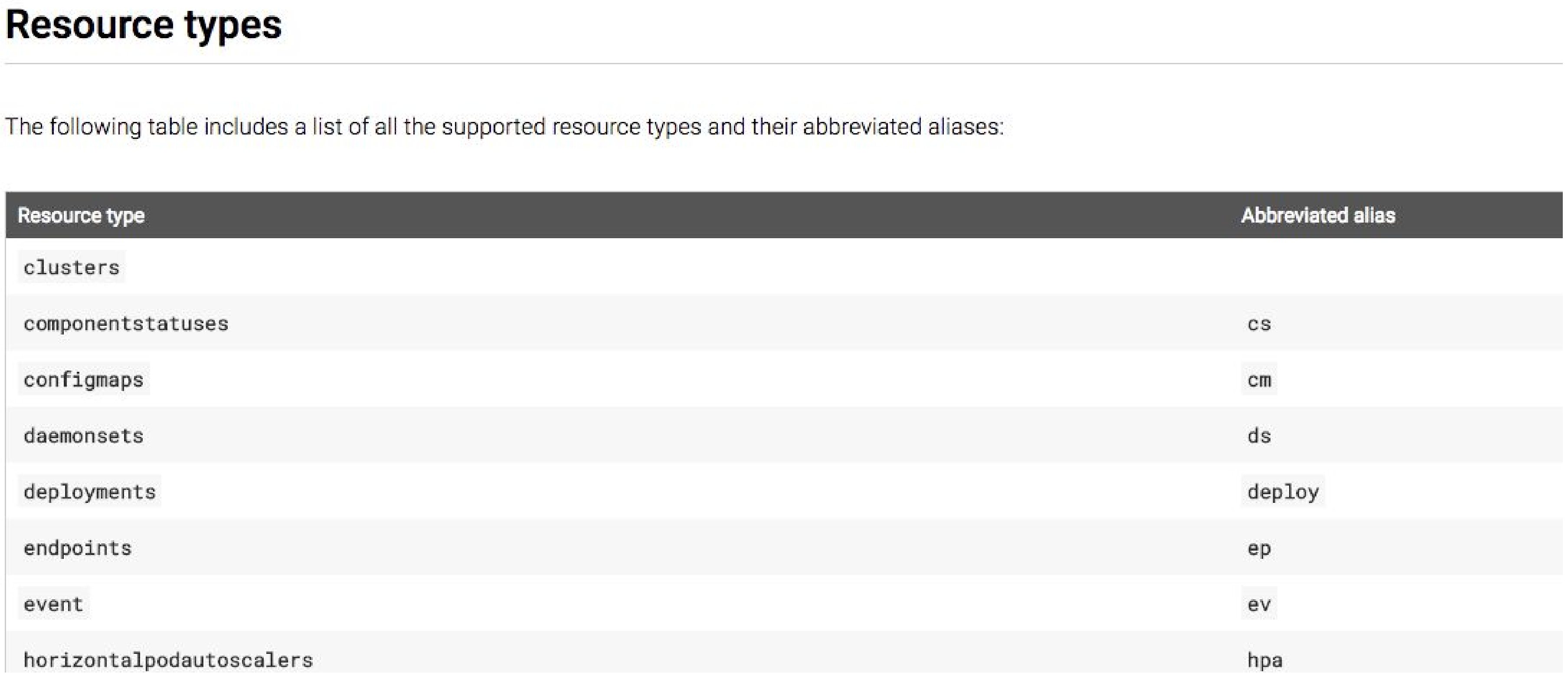
Set which Kubernetes cluster `kubectl` communicates with and modifies configuration information. See Authenticating Across Clusters with kubeconfig documentation for detailed config file information.

```
$ kubectl config view # Show Merged kubeconfig settings.

# use multiple kubeconfig files at the same time and view merged config
$ KUBECONFIG=~/.kube/config:~/.kube/kubconfig2 kubectl config view

# Get the password for the e2e user
$ kubectl config view -o jsonpath='{.users[?(@.name == "e2e")].user.password}'

$ kubectl config current-context              # Display the current-context
$ kubectl config use-context my-cluster-name  # set the default context to my-cluster-name

# add a new cluster to your kubeconf that supports basic auth
$ kubectl config set-credentials kubeuser/foo.kubernetes.com --username=kubeuser --password=kubepassword

# set a context utilizing a specific username and namespace.
$ kubectl config set-context gce --user=cluster-admin --namespace=foo \
  && kubectl config use-context gce
```

# Viewing, Finding Resources



```
# Get commands with basic output
$ kubectl get services                          # List all services in the namespace
$ kubectl get pods --all-namespaces             # List all pods in all namespaces
$ kubectl get pods -o wide                      # List all pods in the namespace, with more details
$ kubectl get deployment my-dep                 # List a particular deployment
$ kubectl get pods --include-uninitialized      # List all pods in the namespace, including uninitialized ones

# Describe commands with verbose output
$ kubectl describe nodes my-node
$ kubectl describe pods my-pod

$ kubectl get services --sort-by=.metadata.name # List Services Sorted by Name
```



# Resource types

The following table includes a list of all the supported resource types and their abbreviated aliases:

| Resource type | Abbreviated alias |
|---|---|
| `clusters` | |
| `componentstatuses` | `cs` |
| `configmaps` | `cm` |
| `daemonsets` | `ds` |
| `deployments` | `deploy` |
| `endpoints` | `ep` |
| `event` | `ev` |
| `horizontalpodautoscalers` | `hpa` |

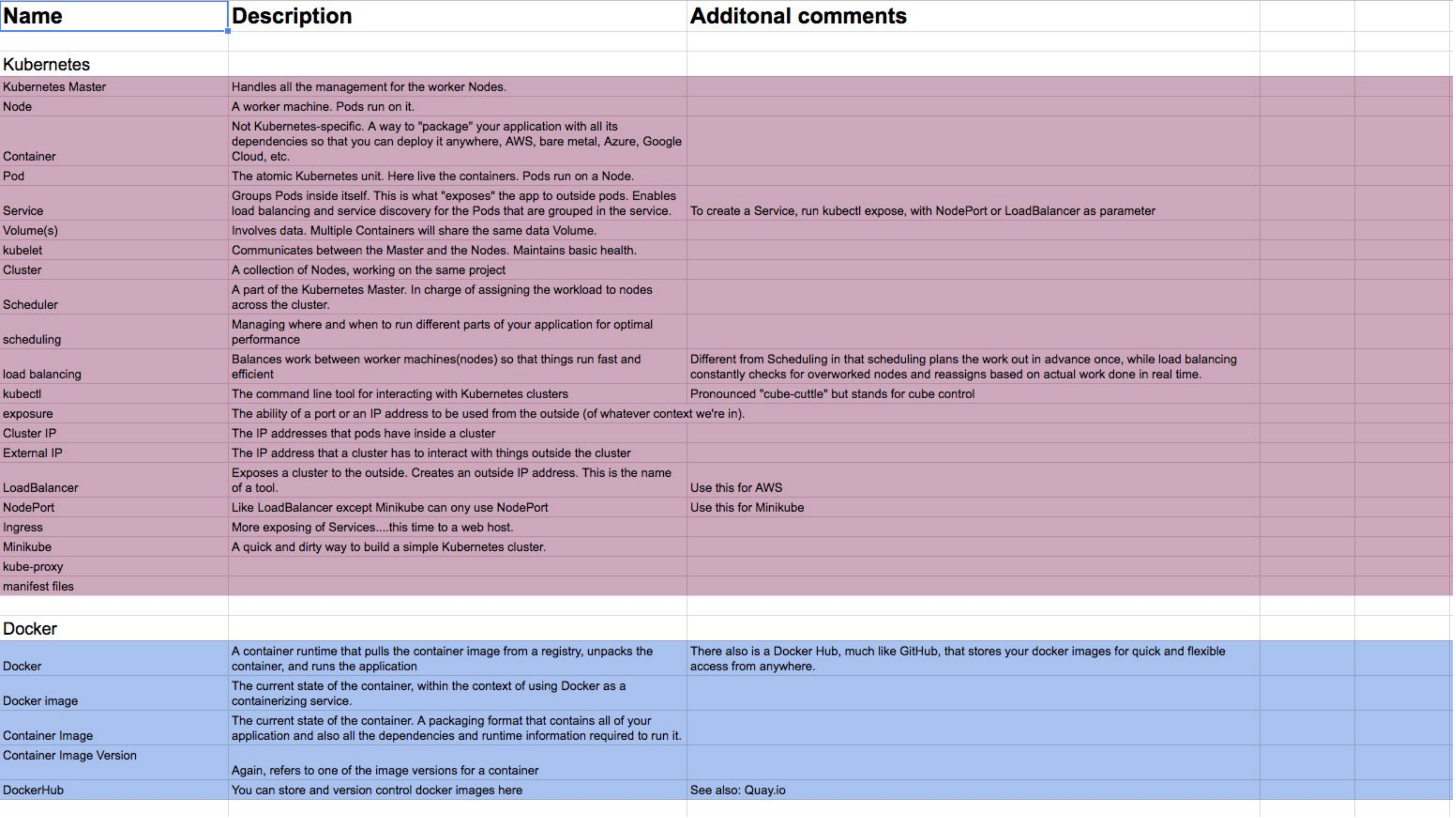
# I’m still confused!

# This is okay.

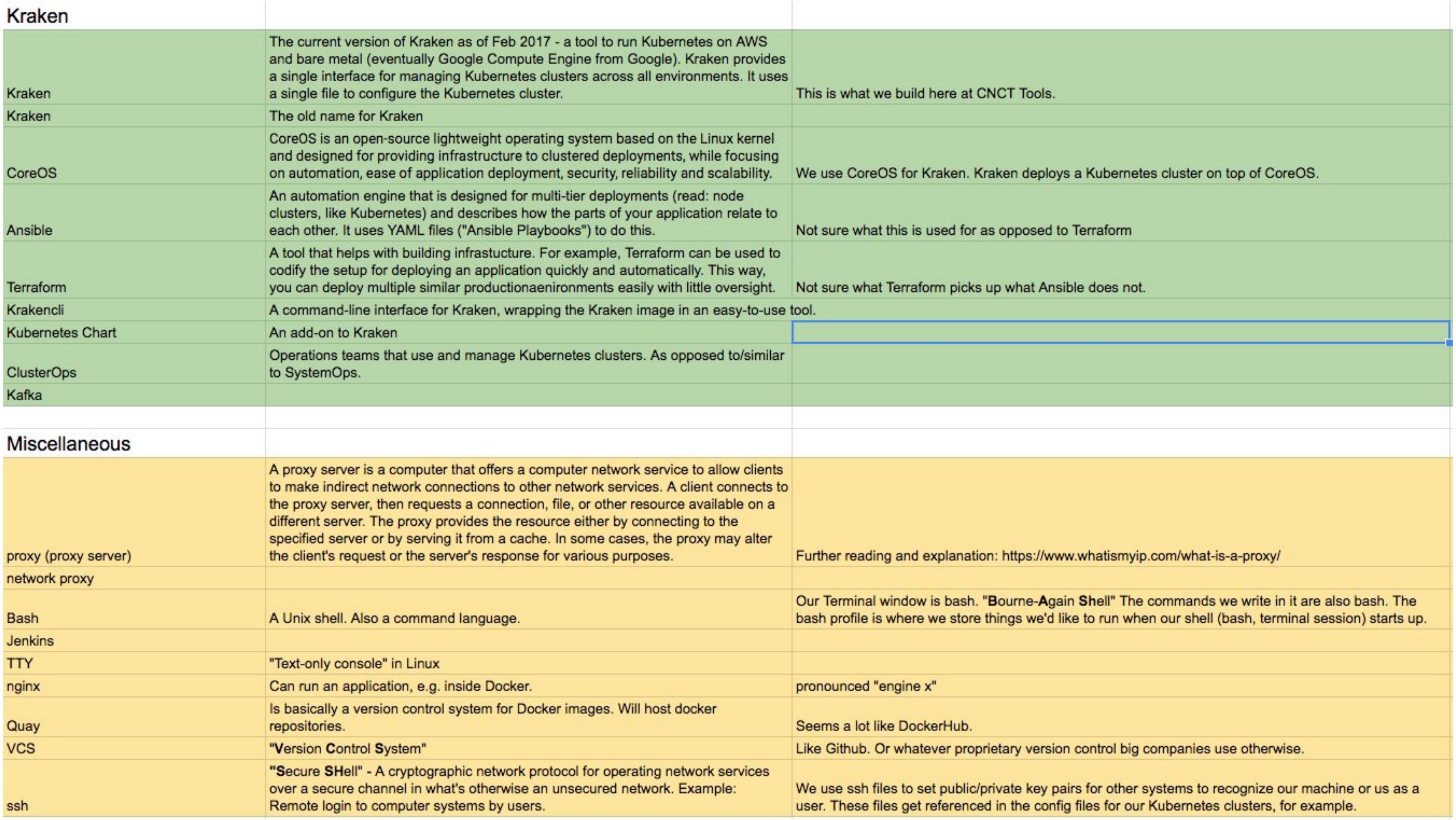
Chances are you are learning lots without knowing it.

# Create a vocabulary list

- Every time you encounter a new concept, write it down
- This will create your very own cheat sheet
- Helps in recognizing what questions to ask
- It is okay if it is wrong or you need to update it
- I literally wrote the definition of bash in mine



| Name | Description | Additonal comments |
|---|---|---|
| Kubernetes | | |
| Kubernetes Master | Handles all the management for the worker Nodes. | |
| Node | A worker machine. Pods run on it. | |
| Container | Not Kubernetes-specific. A way to "package" your application with all its dependencies so that you can deploy it anywhere, AWS, bare metal, Azure, Google Cloud, etc. | |
| Pod | The atomic Kubernetes unit. Here live the containers. Pods run on a Node. | |
| Service | Groups Pods inside itself. This is what "exposes" the app to outside pods. Enables load balancing and service discovery for the Pods that are grouped in the service. | To create a Service, run kubectl expose, with NodePort or LoadBalancer as parameter |
| Volume(s) | Involves data. Multiple Containers will share the same data Volume. | |
| kubelet | Communicates between the Master and the Nodes. Maintains basic health. | |
| Cluster | A collection of Nodes, working on the same project | |
| Scheduler | A part of the Kubernetes Master. In charge of assigning the workload to nodes across the cluster. | |
| scheduling | Managing where and when to run different parts of your application for optimal performance | |
| load balancing | Balances work between worker machines(nodes) so that things run fast and efficient | Different from Scheduling in that scheduling plans the work out in advance once, while load balancing constantly checks for overworked nodes and reassigns based on actual work done in real time. |
| kubectl | The command line tool for interacting with Kubernetes clusters | Pronounced "cube-cuttle" but stands for cube control |
| exposure | The ability of a port or an IP address to be used from the outside (of whatever context we're in). | |
| Cluster IP | The IP addresses that pods have inside a cluster | |
| External IP | The IP address that a cluster has to interact with things outside the cluster | |
| LoadBalancer | Exposes a cluster to the outside. Creates an outside IP address. This is the name of a tool. | Use this for AWS |
| NodePort | Like LoadBalancer except Minikube can ony use NodePort | Use this for Minikube |
| Ingress | More exposing of Services....this time to a web host. | |
| Minikube | A quick and dirty way to build a simple Kubernetes cluster. | |
| kube-proxy | | |
| manifest files | | |
| | | |
| Docker | | |
| Docker | A container runtime that pulls the container image from a registry, unpacks the container, and runs the application | There also is a Docker Hub, much like GitHub, that stores your docker images for quick and flexible access from anywhere. |
| Docker image | The current state of the container, within the context of using Docker as a containerizing service. | |
| Container Image | The current state of the container. A packaging format that contains all of your application and also all the dependencies and runtime information required to run it. | |
| Container Image Version | Again, refers to one of the image versions for a container | |
| DockerHub | You can store and version control docker images here | See also: Quay.io |



| | | |
|---|---|---|
| Kraken | | |
| Kraken | The current version of Kraken as of Feb 2017 - a tool to run Kubernetes on AWS and bare metal (eventually Google Compute Engine from Google). Kraken provides a single interface for managing Kubernetes clusters across all environments. It uses a single file to configure the Kubernetes cluster. | This is what we build here at CNCT Tools. |
| Kraken | The old name for Kraken | |
| CoreOS | CoreOS is an open-source lightweight operating system based on the Linux kernel and designed for providing infrastructure to clustered deployments, while focusing on automation, ease of application deployment, security, reliability and scalability. | We use CoreOS for Kraken. Kraken deploys a Kubernetes cluster on top of CoreOS. |
| Ansible | An automation engine that is designed for multi-tier deployments (read: node clusters, like Kubernetes) and describes how the parts of your application relate to each other. It uses YAML files ("Ansible Playbooks") to do this. | Not sure what this is used for as opposed to Terraform |
| Terraform | A tool that helps with building infrastucture. For example, Terraform can be used to codify the setup for deploying an application quickly and automatically. This way, you can deploy multiple similar productionaenironments easily with little oversight. | Not sure what Terraform picks up what Ansible does not. |
| Krakencli | A command-line interface for Kraken, wrapping the Kraken image in an easy-to-use tool. | |
| Kubernetes Chart | An add-on to Kraken | |
| ClusterOps | Operations teams that use and manage Kubernetes clusters. As opposed to/similar to SystemOps. | |
| Kafka | | |
| | | |
| Miscellaneous | | |
| proxy (proxy server) | A proxy server is a computer that offers a computer network service to allow clients to make indirect network connections to other network services. A client connects to the proxy server, then requests a connection, file, or other resource available on a different server. The proxy provides the resource either by connecting to the specified server or by serving it from a cache. In some cases, the proxy may alter the client's request or the server's response for various purposes. | Further reading and explanation: https://www.whatismyip.com/what-is-a-proxy/ |
| network proxy | | |
| Bash | A Unix shell. Also a command language. | Our Terminal window is bash. "**B**ourne-**A**gain **Sh**ell" The commands we write in it are also bash. The bash profile is where we store things we'd like to run when our shell (bash, terminal session) starts up. |
| Jenkins | | |
| TTY | "Text-only console" in Linux | |
| nginx | Can run an application, e.g. inside Docker. | pronounced "engine x" |
| Quay | Is basically a version control system for Docker images. Will host docker repositories. | Seems a lot like DockerHub. |
| VCS | "**V**ersion **C**ontrol **S**ystem" | Like Github. Or whatever proprietary version control big companies use otherwise. |
| ssh | "**S**ecure **SH**ell" - A cryptographic network protocol for operating network services over a secure channel in what's otherwise an unsecured network. Example: Remote login to computer systems by users. | We use ssh files to set public/private key pairs for other systems to recognize our machine or us as a user. These files get referenced in the config files for our Kubernetes clusters, for example. |

# Try it out yourself

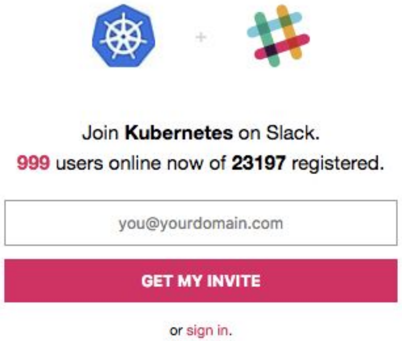
1. Stand up a cluster
2. Write a demo app
3. Put it in a container
4. Deploy it on your cluster

*There are a lot of GUI related tutorials (or paid services) out there, but not many that show you how to do it from running up a cluster all the way to deploying an app. I have not found a good one; if you have, please send me a note!*

# Getting a human to help

# Kubernetes Slack

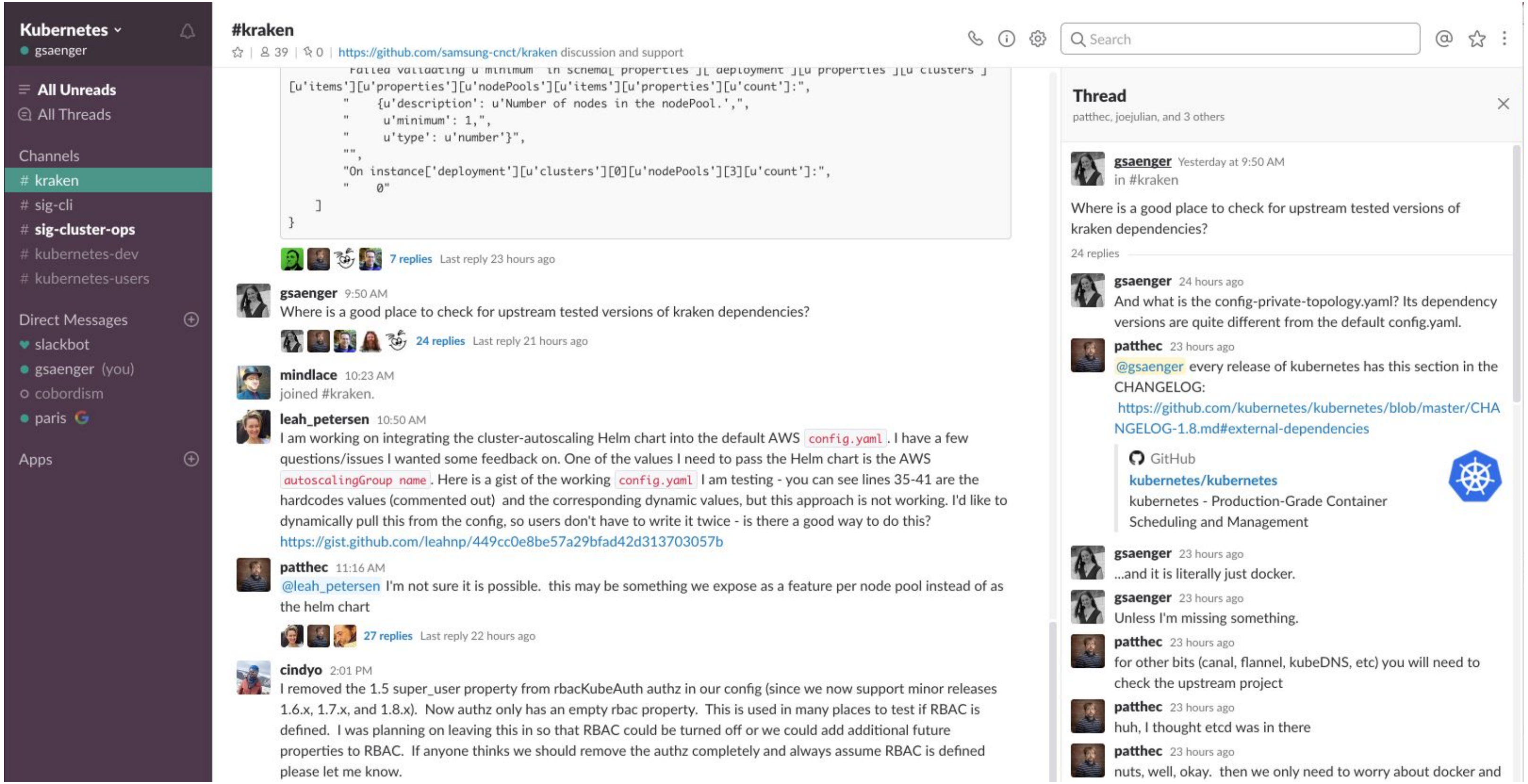
http://slack.kubernetes.io/ for invite



Join **Kubernetes** on Slack.

**999** users online now of **23197** registered.

or sign in.



#kraken

https://github.com/samsung-cnct/kraken discussion and support

```
        Failed validating u'minimum' in schema[u'properties'][u'deployment'][u'properties'][u'clusters']
[u'items'][u'properties'][u'nodePools'][u'items'][u'properties'][u'count']:",
        "    {u'description': u'Number of nodes in the nodePool.',",
        "     u'minimum': 1,",
        "     u'type': u'number'}",
        "",
        "On instance['deployment'][u'clusters'][0][u'nodePools'][3][u'count']:",
        "    0"
    ]
}
```

7 replies Last reply 23 hours ago

**gsaenger** 9:50 AM
Where is a good place to check for upstream tested versions of kraken dependencies?

24 replies Last reply 21 hours ago

**mindlace** 10:23 AM
joined #kraken.

**leah_petersen** 10:50 AM
I am working on integrating the cluster-autoscaling Helm chart into the default AWS `config.yaml`. I have a few questions/issues I wanted some feedback on. One of the values I need to pass the Helm chart is the AWS `autoscalingGroup name`. Here is a gist of the working `config.yaml` I am testing - you can see lines 35-41 are the hardcodes values (commented out) and the corresponding dynamic values, but this approach is not working. I'd like to dynamically pull this from the config, so users don't have to write it twice - is there a good way to do this?
https://gist.github.com/leahnp/449cc0e8be57a29bfad42d313703057b

**patthec** 11:16 AM
@leah_petersen I'm not sure it is possible. this may be something we expose as a feature per node pool instead of as the helm chart

27 replies Last reply 22 hours ago

**cindyo** 2:01 PM
I removed the 1.5 super_user property from rbacKubeAuth authz in our config (since we now support minor releases 1.6.x, 1.7.x, and 1.8.x). Now authz only has an empty rbac property. This is used in many places to test if RBAC is defined. I was planning on leaving this in so that RBAC could be turned off or we could add additional future properties to RBAC. If anyone thinks we should remove the authz completely and always assume RBAC is defined please let me know.

**Thread**

**gsaenger** Yesterday at 9:50 AM
in #kraken

Where is a good place to check for upstream tested versions of kraken dependencies?

24 replies

**gsaenger** 24 hours ago
And what is the config-private-topology.yaml? Its dependency versions are quite different from the default config.yaml.

**patthec** 23 hours ago
@gsaenger every release of kubernetes has this section in the CHANGELOG:
https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG-1.8.md#external-dependencies

GitHub
kubernetes/kubernetes
kubernetes - Production-Grade Container Scheduling and Management

**gsaenger** 23 hours ago
...and it is literally just docker.

**gsaenger** 23 hours ago
Unless I'm missing something.

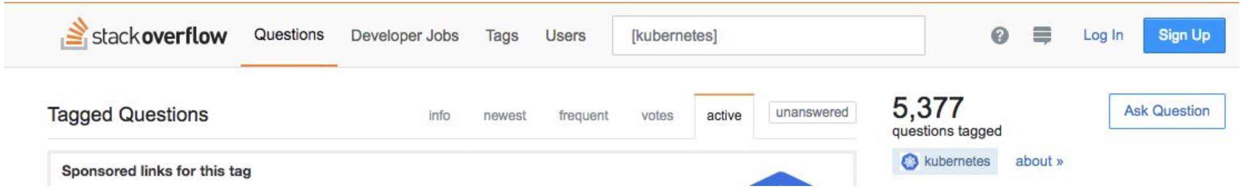
**patthec** 23 hours ago
for other bits (canal, flannel, kubeDNS, etc) you will need to check the upstream project

**patthec** 23 hours ago
huh, I thought etcd was in there

**patthec** 23 hours ago
nuts, well, okay. then we only need to worry about docker and

# Stack Overflow

https://stackoverflow.com/questions/tagged/kubernetes

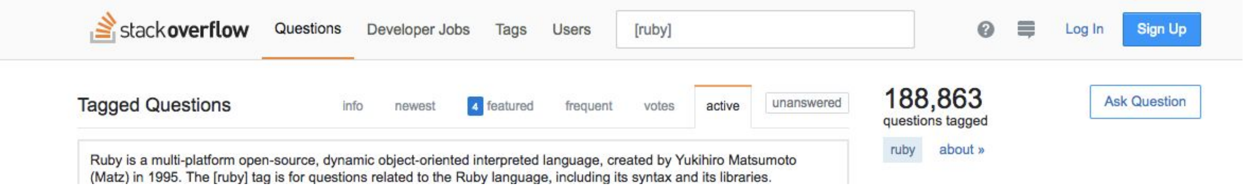


5,377
questions tagged

Sponsored links for this tag

Stackoverflow can use your help!

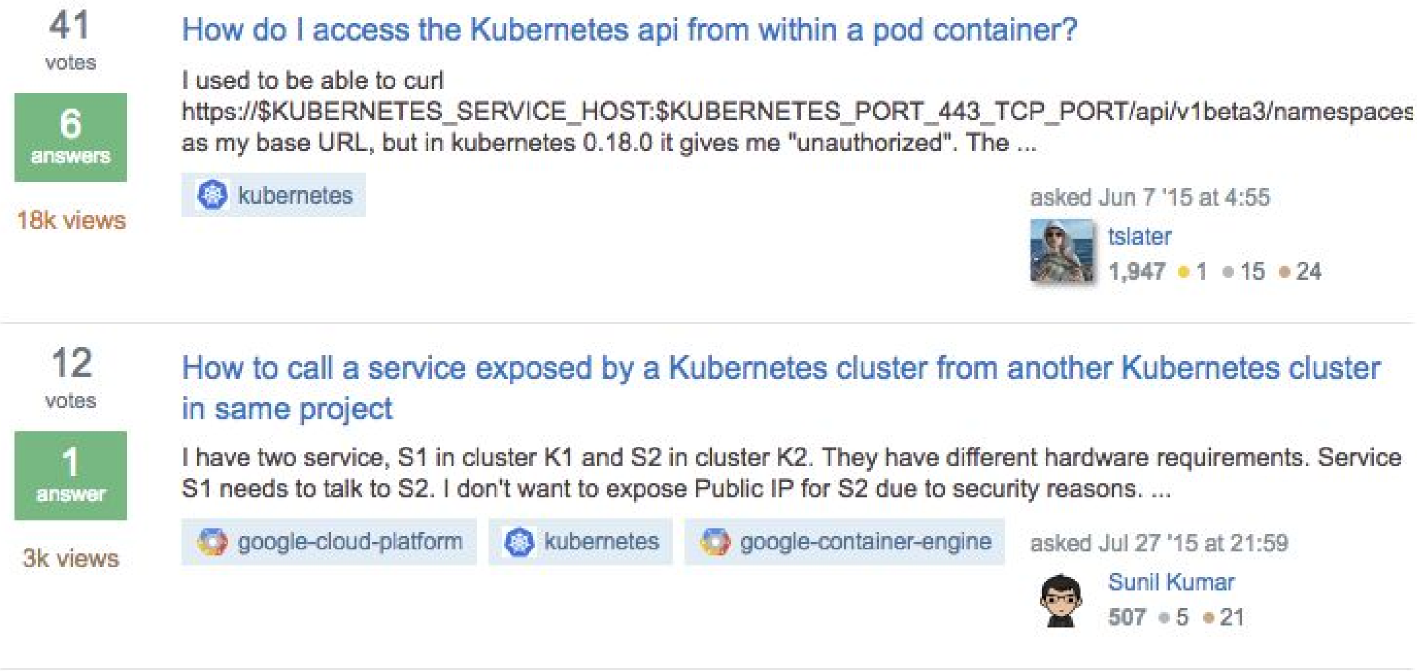
By comparison...



188,863
questions tagged

Ruby is a multi-platform open-source, dynamic object-oriented interpreted language, created by Yukihiro Matsumoto (Matz) in 1995. The [ruby] tag is for questions related to the Ruby language, including its syntax and its libraries.

# Looking for frequent questions...



41
votes

6
answers

18k views

How do I access the Kubernetes api from within a pod container?

I used to be able to curl
https://$KUBERNETES_SERVICE_HOST:$KUBERNETES_PORT_443_TCP_PORT/api/v1beta3/namespaces
as my base URL, but in kubernetes 0.18.0 it gives me "unauthorized". The ...

kubernetes

asked Jun 7 '15 at 4:55
tslater
1,947 ● 1 ● 15 ● 24

12
votes

1
answer

3k views

How to call a service exposed by a Kubernetes cluster from another Kubernetes cluster in same project

I have two service, S1 in cluster K1 and S2 in cluster K2. They have different hardware requirements. Service S1 needs to talk to S2. I don't want to expose Public IP for S2 due to security reasons. ...

google-cloud-platform   kubernetes   google-container-engine

asked Jul 27 '15 at 21:59
Sunil Kumar
507 ● 5 ● 21

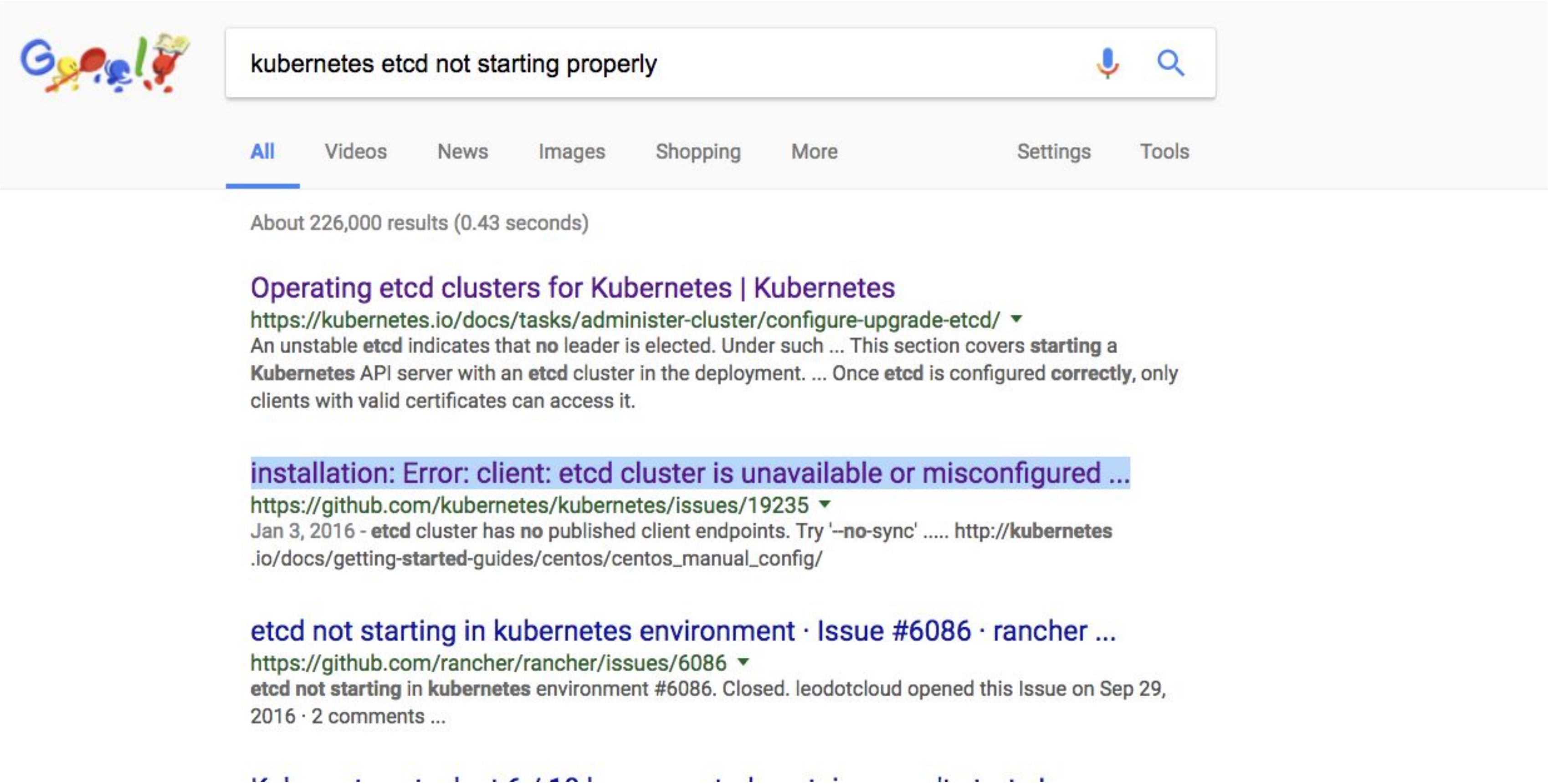
# Kubernetes Office Hours

https://github.com/kubernetes/community/blob/master/community/office-hours.md

Once-a-month live-stream on Youtube

Ask your question on Stackoverflow, and have it addressed in Office Hours!

# Sometimes, the solution is found upstream



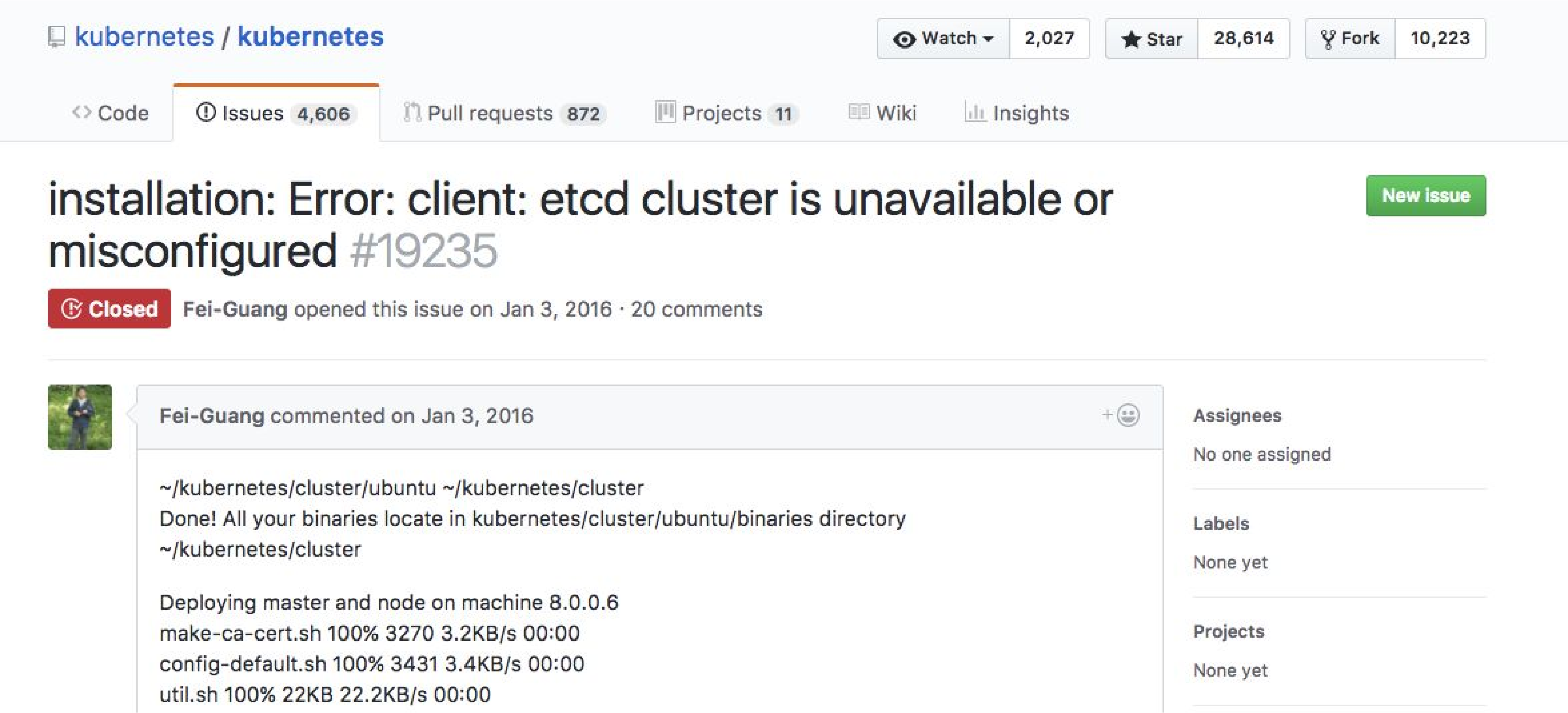
kubernetes etcd not starting properly

About 226,000 results (0.43 seconds)

Operating etcd clusters for Kubernetes | Kubernetes
https://kubernetes.io/docs/tasks/administer-cluster/configure-upgrade-etcd/
An unstable **etcd** indicates that **no** leader is elected. Under such ... This section covers **starting** a **Kubernetes** API server with an **etcd** cluster in the deployment. ... Once **etcd** is configured **correctly**, only clients with valid certificates can access it.

installation: Error: client: etcd cluster is unavailable or misconfigured ...
https://github.com/kubernetes/kubernetes/issues/19235
Jan 3, 2016 - **etcd** cluster has **no** published client endpoints. Try '**--no-sync**' ..... http://**kubernetes** .io/docs/getting-**started**-guides/centos/centos_manual_config/

etcd not starting in kubernetes environment · Issue #6086 · rancher ...
https://github.com/rancher/rancher/issues/6086
**etcd not starting** in **kubernetes** environment #6086. Closed. leodotcloud opened this Issue on Sep 29, 2016 · 2 comments ...



# installation: Error: client: etcd cluster is unavailable or misconfigured #19235

**Fei-Guang** opened this issue on Jan 3, 2016 · 20 comments

**Fei-Guang** commented on Jan 3, 2016

```
~/kubernetes/cluster/ubuntu ~/kubernetes/cluster
Done! All your binaries locate in kubernetes/cluster/ubuntu/binaries directory
~/kubernetes/cluster

Deploying master and node on machine 8.0.0.6
make-ca-cert.sh 100% 3270 3.2KB/s 00:00
config-default.sh 100% 3431 3.4KB/s 00:00
util.sh 100% 22KB 22.2KB/s 00:00
```

**Assignees**

No one assigned

**Labels**

None yet

**Projects**

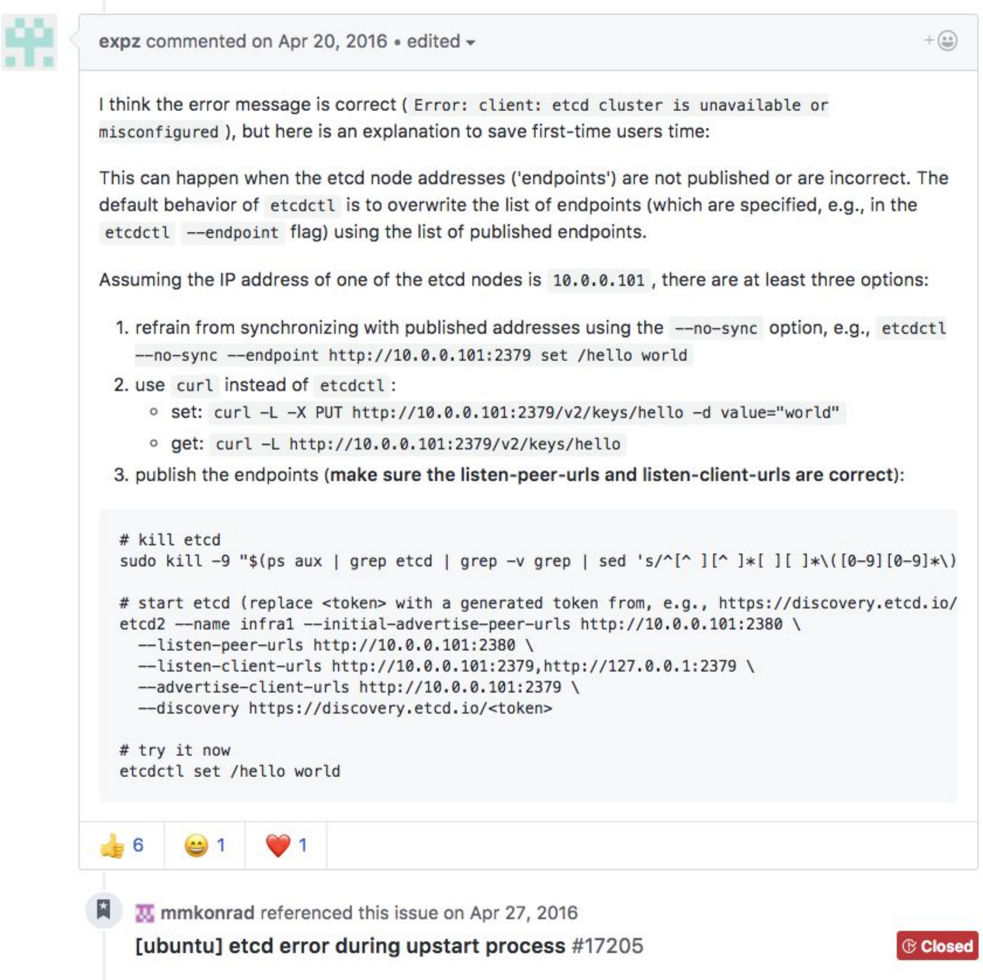
None yet

Side note:

Generally, github is not the place for individual troubleshooting.

However, it is good to check if yours is a known issue. Workarounds or fixes may be available.

Remember - people are friendly and will help you out.



**expz** commented on Apr 20, 2016 • edited ▾

I think the error message is correct ( `Error: client: etcd cluster is unavailable or misconfigured` ), but here is an explanation to save first-time users time:

This can happen when the etcd node addresses ('endpoints') are not published or are incorrect. The default behavior of `etcdctl` is to overwrite the list of endpoints (which are specified, e.g., in the `etcdctl` `--endpoint` flag) using the list of published endpoints.

Assuming the IP address of one of the etcd nodes is `10.0.0.101`, there are at least three options:

1. refrain from synchronizing with published addresses using the `--no-sync` option, e.g., `etcdctl --no-sync --endpoint http://10.0.0.101:2379 set /hello world`
2. use `curl` instead of `etcdctl`:
   - set: `curl -L -X PUT http://10.0.0.101:2379/v2/keys/hello -d value="world"`
   - get: `curl -L http://10.0.0.101:2379/v2/keys/hello`
3. publish the endpoints (**make sure the listen-peer-urls and listen-client-urls are correct**):

```
# kill etcd
sudo kill -9 "$(ps aux | grep etcd | grep -v grep | sed 's/^[^ ][^ ]*[ ][ ]*\([0-9][0-9]*\)

# start etcd (replace <token> with a generated token from, e.g., https://discovery.etcd.io/
etcd2 --name infra1 --initial-advertise-peer-urls http://10.0.0.101:2380 \
  --listen-peer-urls http://10.0.0.101:2380 \
  --listen-client-urls http://10.0.0.101:2379,http://127.0.0.1:2379 \
  --advertise-client-urls http://10.0.0.101:2379 \
  --discovery https://discovery.etcd.io/<token>

# try it now
etcdctl set /hello world
```

👍 6 😄 1 ❤️ 1

**mmkonrad** referenced this issue on Apr 27, 2016

**[ubuntu] etcd error during upstart process** #17205

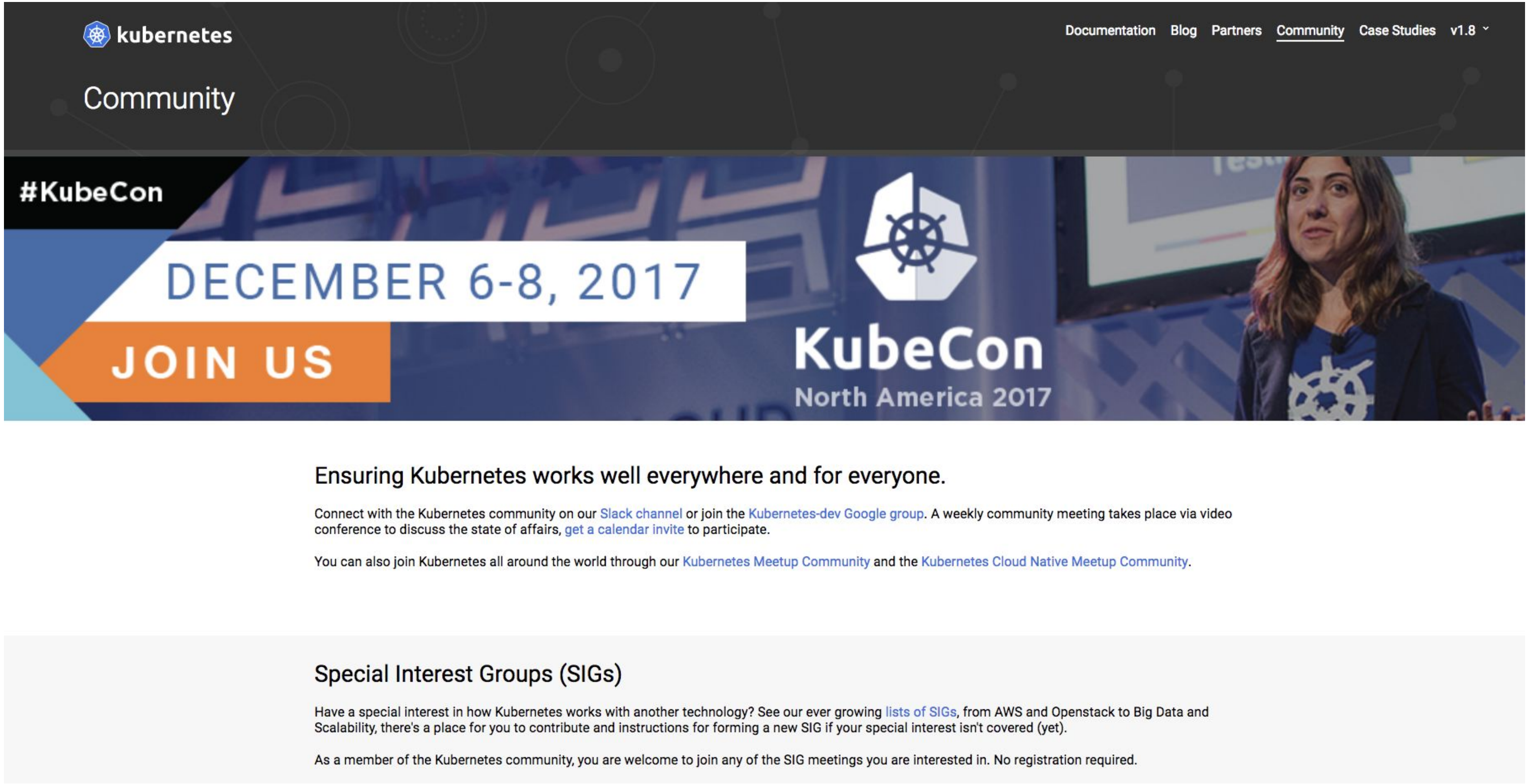
# Finding an actual human

Success! You are here, amongst lots of humans who use Kubernetes.

But also....

# Kubernetes Community

https://kubernetes.io/community/



Community

## Ensuring Kubernetes works well everywhere and for everyone.

Connect with the Kubernetes community on our Slack channel or join the Kubernetes-dev Google group. A weekly community meeting takes place via video conference to discuss the state of affairs, get a calendar invite to participate.

You can also join Kubernetes all around the world through our Kubernetes Meetup Community and the Kubernetes Cloud Native Meetup Community.

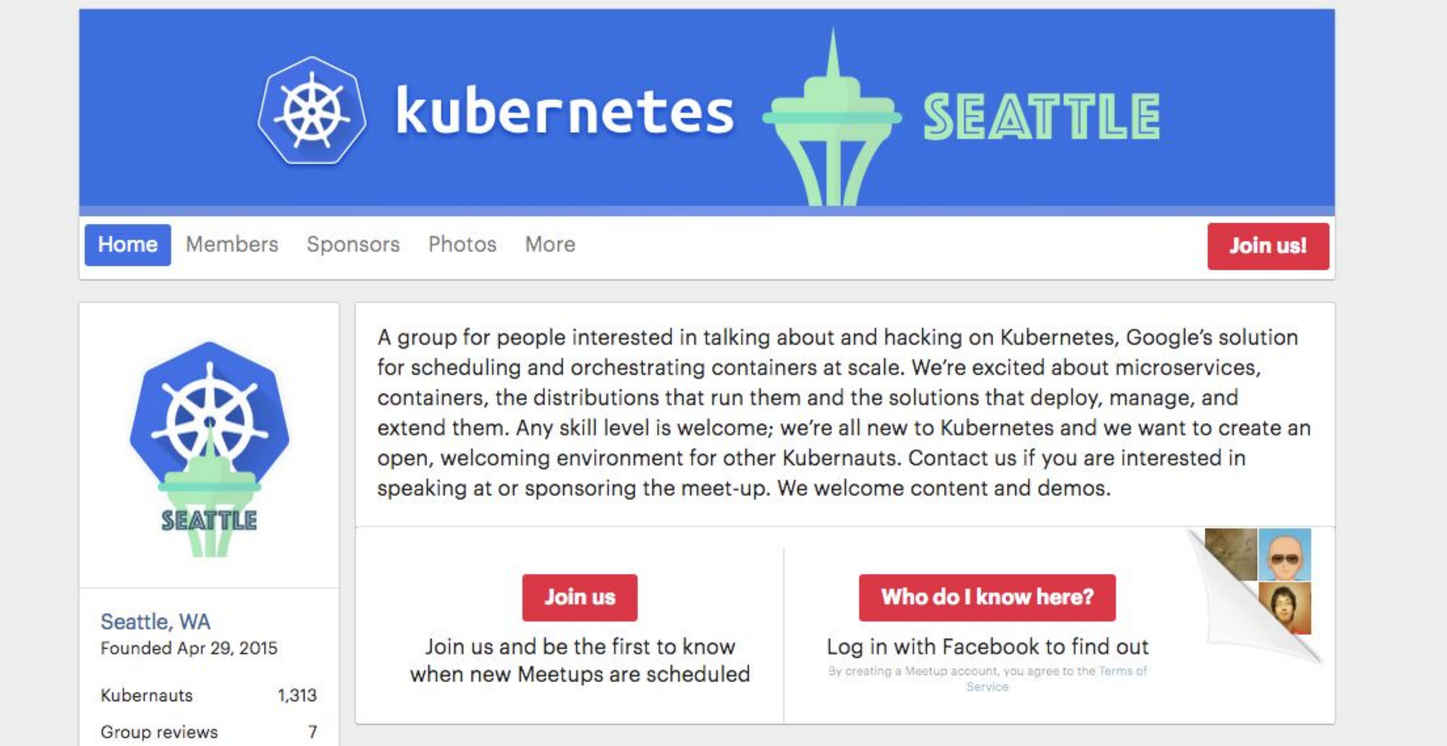
## Special Interest Groups (SIGs)

Have a special interest in how Kubernetes works with another technology? See our ever growing lists of SIGs, from AWS and Openstack to Big Data and Scalability, there's a place for you to contribute and instructions for forming a new SIG if your special interest isn't covered (yet).

As a member of the Kubernetes community, you are welcome to join any of the SIG meetings you are interested in. No registration required.

# Attend a meetup

https://www.meetup.com/topics/kubernetes/



A group for people interested in talking about and hacking on Kubernetes, Google's solution for scheduling and orchestrating containers at scale. We're excited about microservices, containers, the distributions that run them and the solutions that deploy, manage, and extend them. Any skill level is welcome; we're all new to Kubernetes and we want to create an open, welcoming environment for other Kubernauts. Contact us if you are interested in speaking at or sponsoring the meet-up. We welcome content and demos.

Join us and be the first to know when new Meetups are scheduled

Log in with Facebook to find out

By creating a Meetup account, you agree to the Terms of Service

Seattle, WA
Founded Apr 29, 2015

| | |
|---|---|
| Kubernauts | 1,313 |
| Group reviews | 7 |

SAMSUNG SDS SAMSUNG

# No Meetup nearby? Start one!

- Learn which companies/individuals use K8s
- Connect with them
- Remember: People are mostly friendly!
- Meetup Starter Help: Contact Paris Pittman
  parispittman@google.com

# Go old school

Ask someone if they’re willing to meet and get you started.

All they can say is no (but they probably won’t).

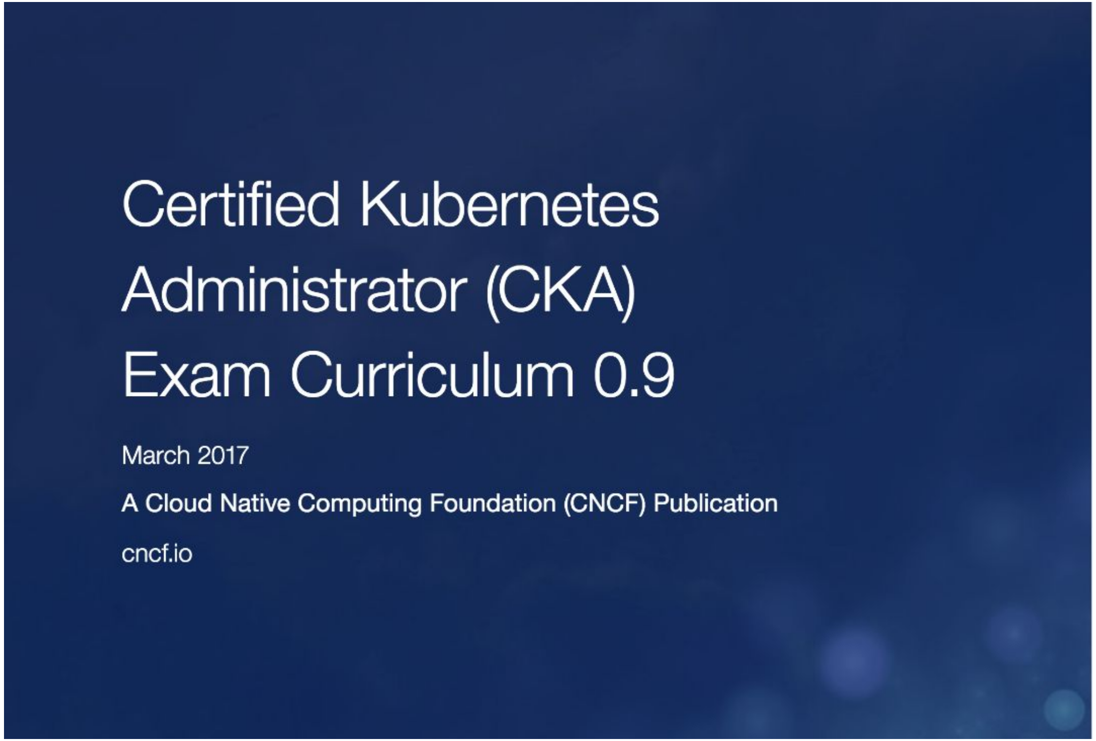
# Digging deeper

# The Certified Kubernetes Administrator Exam

*"This was undoubtedly the best way to learn about Kubernetes in depth"*
- my coworker, (ab)using the exam as his personal study guide

# Exam curriculum

https://github.com/cncf/curriculum/blob/master/certified_kubernetes_administrator_exam_V0.9.pdf



Certified Kubernetes Administrator (CKA) Exam Curriculum 0.9

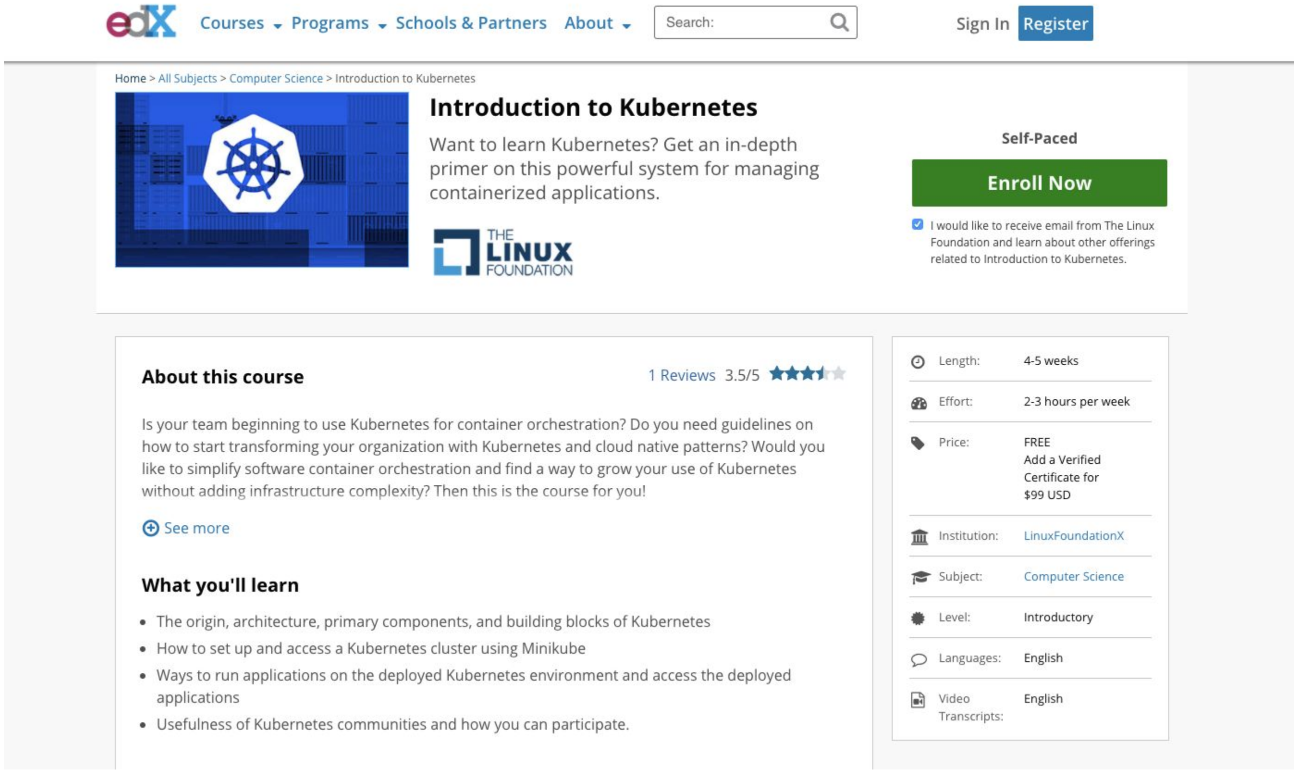
March 2017

A Cloud Native Computing Foundation (CNCF) Publication

cncf.io

# edX course

https://www.edx.org/course/introduction-kubernetes-linuxfoundationx-lfs158x



Home > All Subjects > Computer Science > Introduction to Kubernetes

## Introduction to Kubernetes

Want to learn Kubernetes? Get an in-depth primer on this powerful system for managing containerized applications.

THE LINUX FOUNDATION

**Self-Paced**

☑ I would like to receive email from The Linux Foundation and learn about other offerings related to Introduction to Kubernetes.

### About this course

1 Reviews 3.5/5 ★★★½☆

Is your team beginning to use Kubernetes for container orchestration? Do you need guidelines on how to start transforming your organization with Kubernetes and cloud native patterns? Would you like to simplify software container orchestration and find a way to grow your use of Kubernetes without adding infrastructure complexity? Then this is the course for you!

⊕ See more

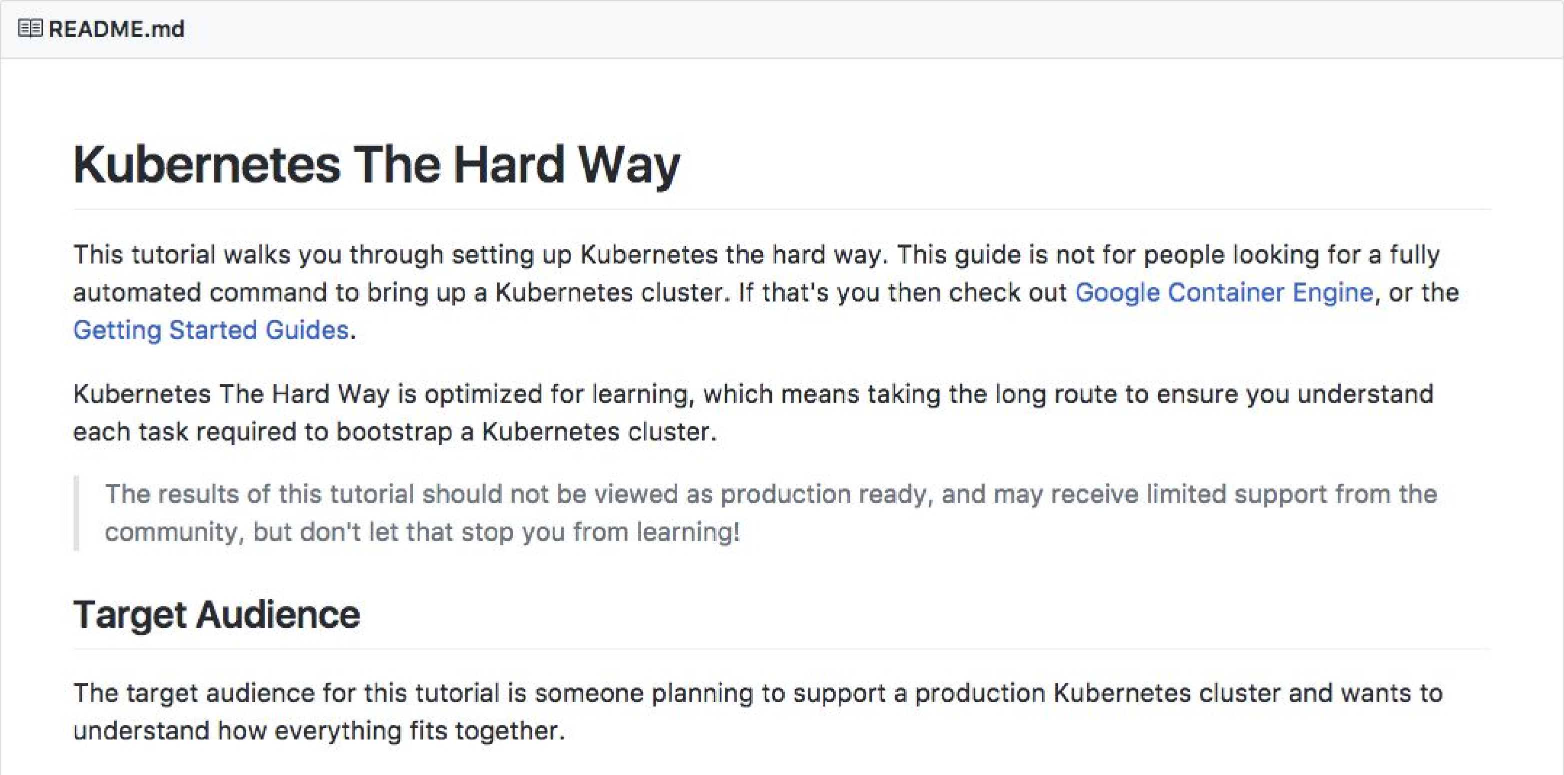
### What you'll learn

- The origin, architecture, primary components, and building blocks of Kubernetes
- How to set up and access a Kubernetes cluster using Minikube
- Ways to run applications on the deployed Kubernetes environment and access the deployed applications
- Usefulness of Kubernetes communities and how you can participate.

| | |
|---|---|
| Length: | 4-5 weeks |
| Effort: | 2-3 hours per week |
| Price: | FREE<br>Add a Verified Certificate for $99 USD |
| Institution: | LinuxFoundationX |
| Subject: | Computer Science |
| Level: | Introductory |
| Languages: | English |
| Video Transcripts: | English |

# Kubernetes The Hard Way

https://github.com/kelseyhightower/kubernetes-the-hard-way



## Kubernetes The Hard Way

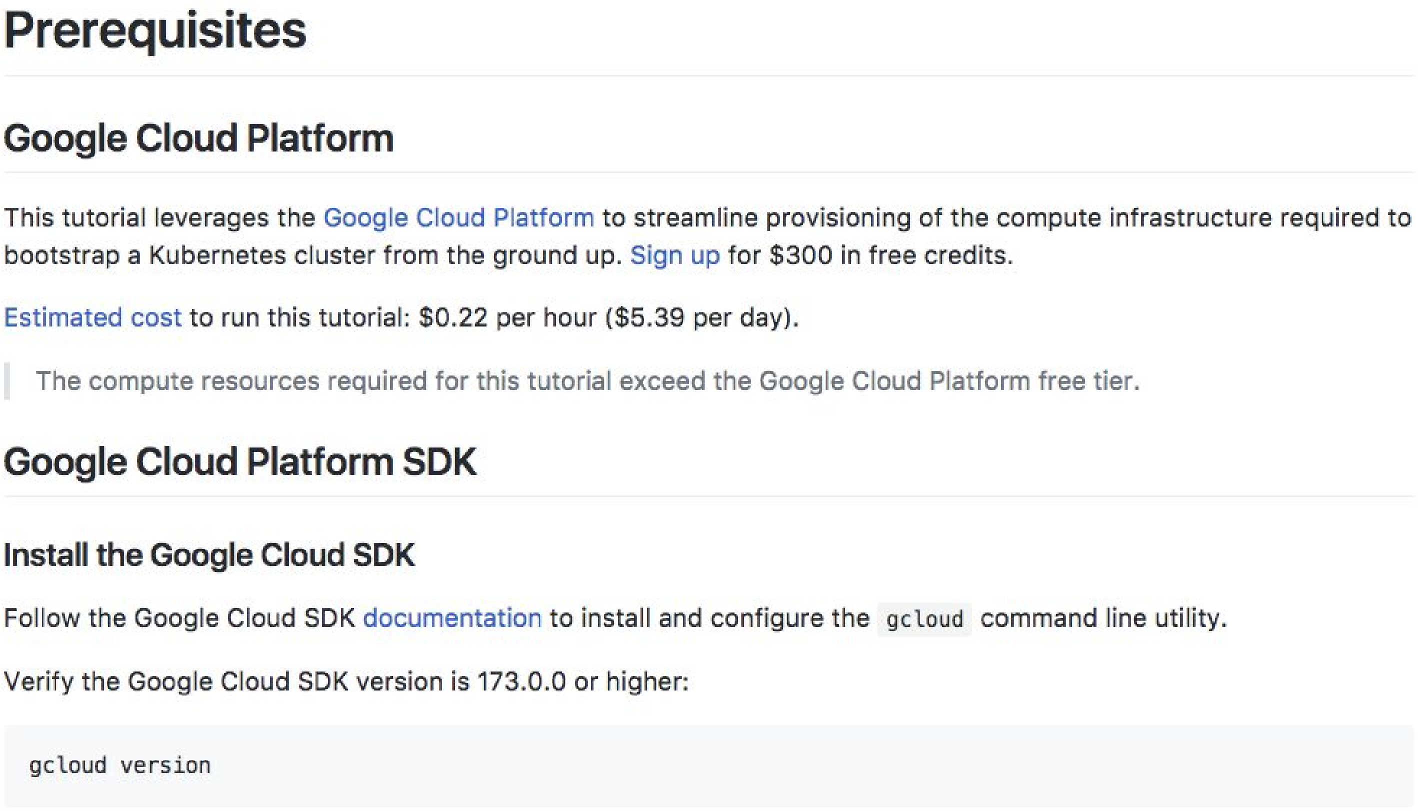
This tutorial walks you through setting up Kubernetes the hard way. This guide is not for people looking for a fully automated command to bring up a Kubernetes cluster. If that's you then check out Google Container Engine, or the Getting Started Guides.

Kubernetes The Hard Way is optimized for learning, which means taking the long route to ensure you understand each task required to bootstrap a Kubernetes cluster.

> The results of this tutorial should not be viewed as production ready, and may receive limited support from the community, but don't let that stop you from learning!

## Target Audience

The target audience for this tutorial is someone planning to support a production Kubernetes cluster and wants to understand how everything fits together.



# Prerequisites

## Google Cloud Platform

This tutorial leverages the Google Cloud Platform to streamline provisioning of the compute infrastructure required to bootstrap a Kubernetes cluster from the ground up. Sign up for $300 in free credits.

Estimated cost to run this tutorial: $0.22 per hour ($5.39 per day).

> The compute resources required for this tutorial exceed the Google Cloud Platform free tier.

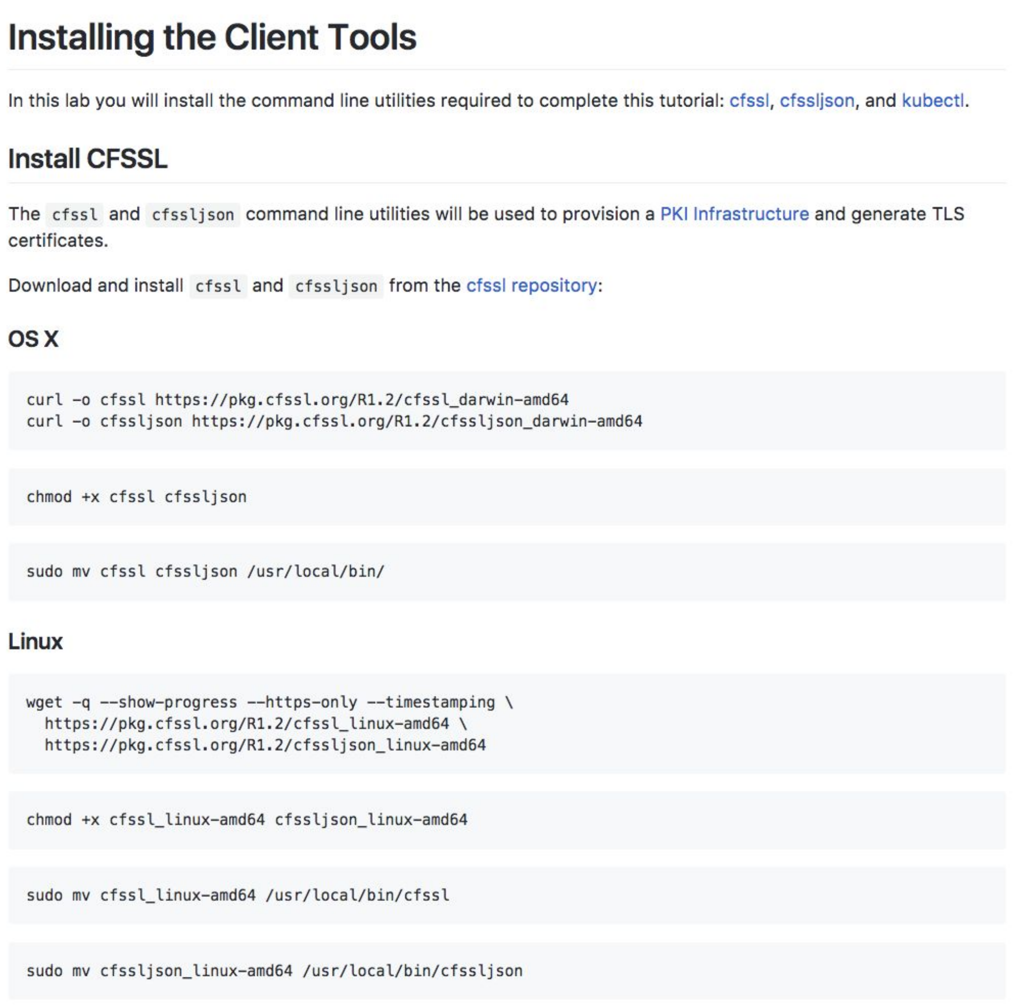
## Google Cloud Platform SDK

### Install the Google Cloud SDK

Follow the Google Cloud SDK documentation to install and configure the `gcloud` command line utility.

Verify the Google Cloud SDK version is 173.0.0 or higher:

```
gcloud version
```



# Installing the Client Tools

In this lab you will install the command line utilities required to complete this tutorial: cfssl, cfssljson, and kubectl.

## Install CFSSL

The `cfssl` and `cfssljson` command line utilities will be used to provision a PKI Infrastructure and generate TLS certificates.

Download and install `cfssl` and `cfssljson` from the cfssl repository:

### OS X

```
curl -o cfssl https://pkg.cfssl.org/R1.2/cfssl_darwin-amd64
curl -o cfssljson https://pkg.cfssl.org/R1.2/cfssljson_darwin-amd64
```

```
chmod +x cfssl cfssljson
```

```
sudo mv cfssl cfssljson /usr/local/bin/
```

### Linux

```
wget -q --show-progress --https-only --timestamping \
  https://pkg.cfssl.org/R1.2/cfssl_linux-amd64 \
  https://pkg.cfssl.org/R1.2/cfssljson_linux-amd64
```

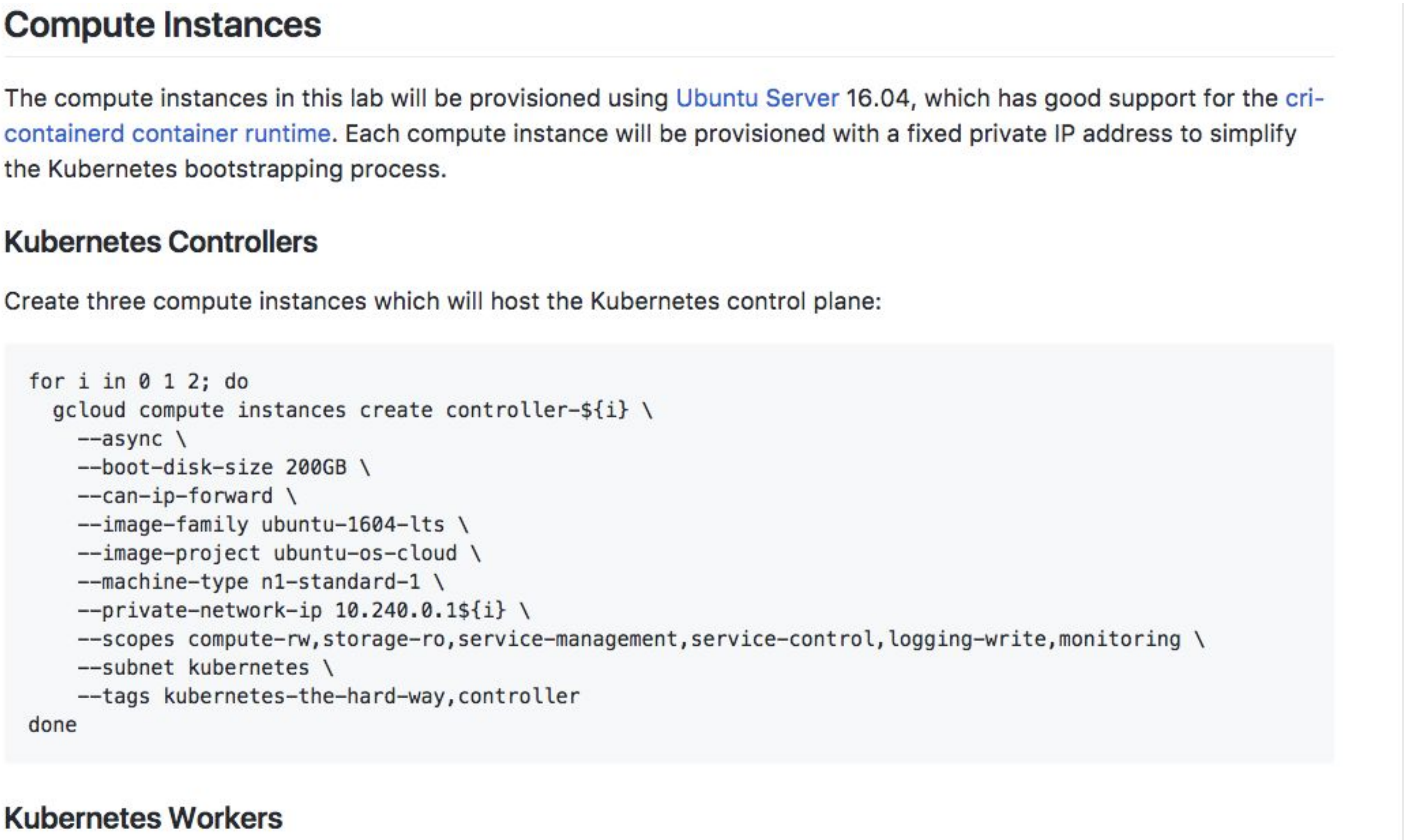
```
chmod +x cfssl_linux-amd64 cfssljson_linux-amd64
```

```
sudo mv cfssl_linux-amd64 /usr/local/bin/cfssl
```

```
sudo mv cfssljson_linux-amd64 /usr/local/bin/cfssljson
```

The installfest...

# Bootstrapping your own cluster parts!



## Compute Instances

The compute instances in this lab will be provisioned using Ubuntu Server 16.04, which has good support for the cri-containerd container runtime. Each compute instance will be provisioned with a fixed private IP address to simplify the Kubernetes bootstrapping process.

### Kubernetes Controllers

Create three compute instances which will host the Kubernetes control plane:

```
for i in 0 1 2; do
  gcloud compute instances create controller-${i} \
    --async \
    --boot-disk-size 200GB \
    --can-ip-forward \
    --image-family ubuntu-1604-lts \
    --image-project ubuntu-os-cloud \
    --machine-type n1-standard-1 \
    --private-network-ip 10.240.0.1${i} \
    --scopes compute-rw,storage-ro,service-management,service-control,logging-write,monitoring \
    --subnet kubernetes \
    --tags kubernetes-the-hard-way,controller
done
```

### Kubernetes Workers

# Trouble with Linux

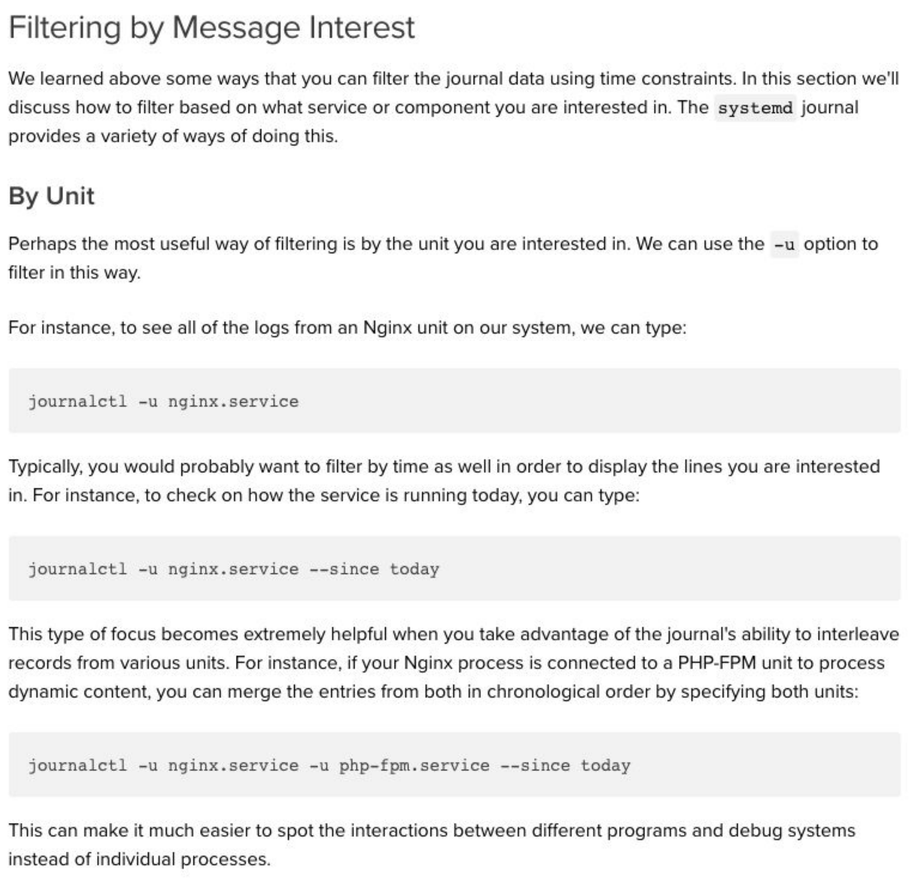
Containers and pods run with Linux, as shown in Kubernetes The Hard Way.

If you have never dealt with Linux, it can be difficult to find helpful articles to get started.

Fortunately, many more people know about Linux than know about Kubernetes.

These articles really helped me:
https://www.digitalocean.com/community/tutorials/how-to-use-journalctl-to-view-and-manipulate-systemd-logs
https://www.digitalocean.com/community/tutorials/systemd-essentials-working-with-services-units-and-the-journal

# Journalctl for logs



## Filtering by Message Interest

We learned above some ways that you can filter the journal data using time constraints. In this section we'll discuss how to filter based on what service or component you are interested in. The `systemd` journal provides a variety of ways of doing this.

### By Unit

Perhaps the most useful way of filtering is by the unit you are interested in. We can use the `-u` option to filter in this way.

For instance, to see all of the logs from an Nginx unit on our system, we can type:

```
journalctl -u nginx.service
```

Typically, you would probably want to filter by time as well in order to display the lines you are interested in. For instance, to check on how the service is running today, you can type:

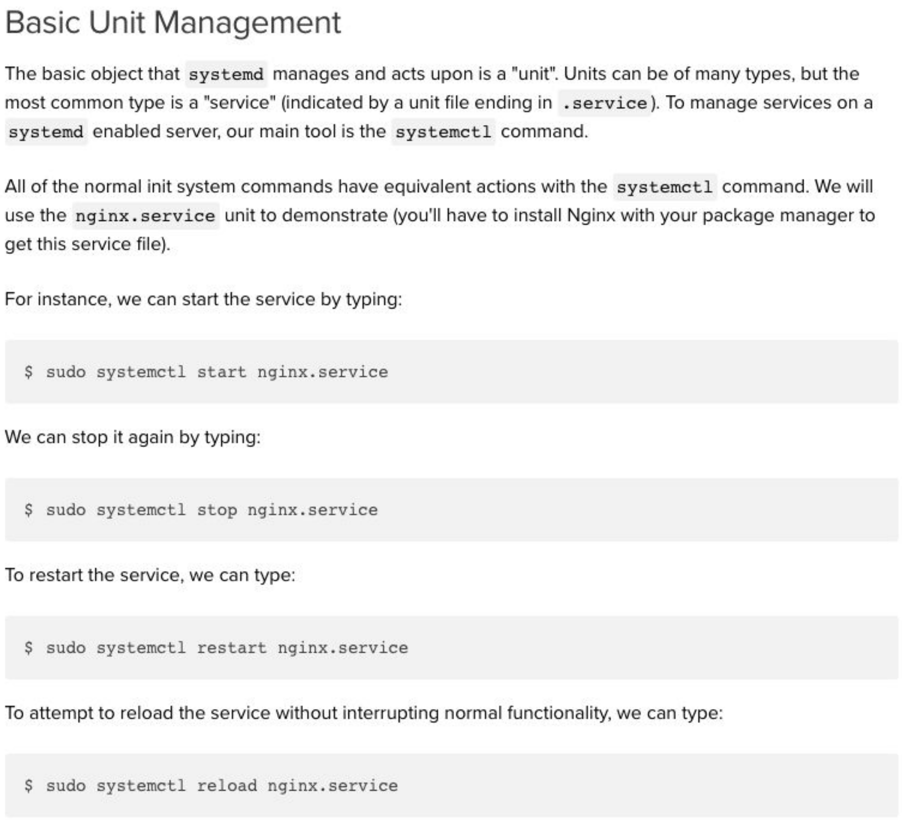
```
journalctl -u nginx.service --since today
```

This type of focus becomes extremely helpful when you take advantage of the journal's ability to interleave records from various units. For instance, if your Nginx process is connected to a PHP-FPM unit to process dynamic content, you can merge the entries from both in chronological order by specifying both units:

```
journalctl -u nginx.service -u php-fpm.service --since today
```

This can make it much easier to spot the interactions between different programs and debug systems instead of individual processes.

# And systemctl for units.



## Basic Unit Management

The basic object that `systemd` manages and acts upon is a "unit". Units can be of many types, but the most common type is a "service" (indicated by a unit file ending in `.service`). To manage services on a `systemd` enabled server, our main tool is the `systemctl` command.

All of the normal init system commands have equivalent actions with the `systemctl` command. We will use the `nginx.service` unit to demonstrate (you'll have to install Nginx with your package manager to get this service file).

For instance, we can start the service by typing:

```
$ sudo systemctl start nginx.service
```

We can stop it again by typing:

```
$ sudo systemctl stop nginx.service
```

To restart the service, we can type:

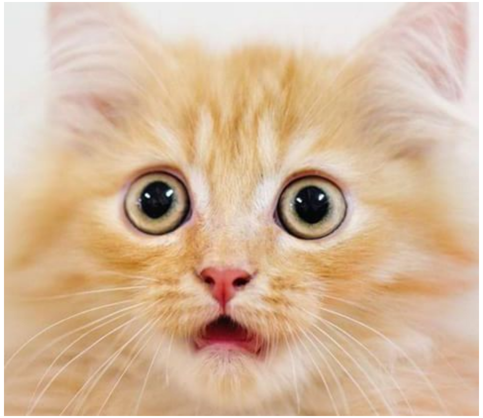
```
$ sudo systemctl restart nginx.service
```

To attempt to reload the service without interrupting normal functionality, we can type:

```
$ sudo systemctl reload nginx.service
```

# So I took the CKA exam...

...the first month of my being a full time employee at Samsung.



It took four hours, and if you failed the first time, there was a retake option.

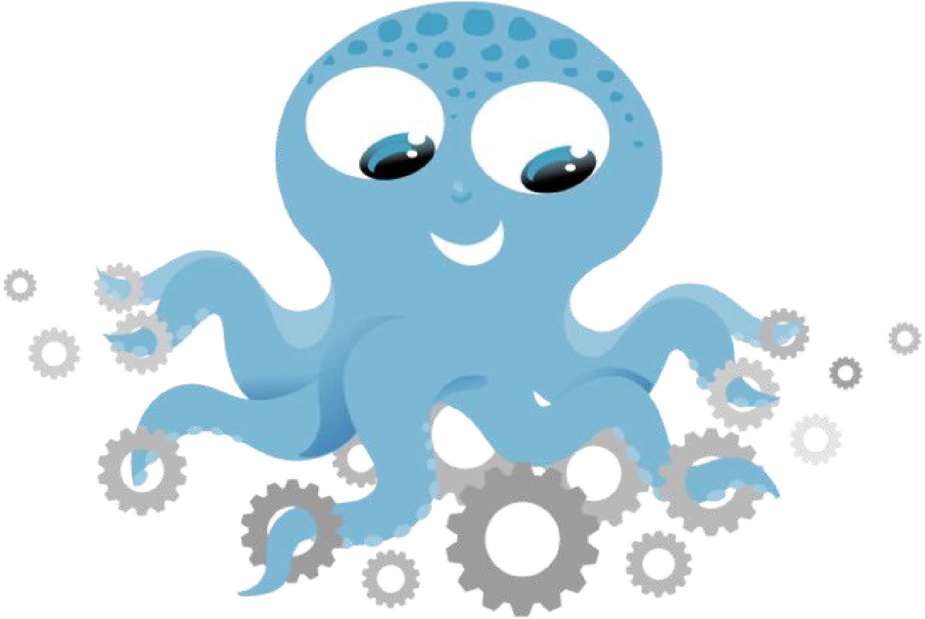
This is what that felt like...

The ultimate result….

# So what do I do currently?

# Kraken

https://github.com/samsung-cnct/kraken



SAMSUNG SDS SAMSUNG

# Kraken

## Samsung SDS America

I really liked my internship, and the feeling was mutual, so now I work there!

We build a tool called Kraken, which creates Kubernetes clusters made to order.

I enjoy making Kubernetes easier for other people.

SAMSUNG SDS SAMSUNG

# Contributing to Kubernetes

Just a few thoughts, since this could easily fill up its own talk

Github community page for getting started (and possibly overwhelmed) https://github.com/kubernetes/community

We want everyone to contribute and have a good time. The Contributor Guide is currently underway to improve newcomers’ experience. If you notice something missing you’d like to see, file an issue, or submit a PR!

# Questions?

# Guinevere Saenger

Software Engineer, Samsung SDS America

https://github.com/guineveresaenger

@guincodes

guineveresaenger@gmail.com

@gsaenger

https://www.linkedin.com/in/guinevere-saenger/

SAMSUNG SDS SAMSUNG

# Reference Links

Getting Started:

https://kubernetes.io/docs/tutorials/kubernetes-basics/

https://www.udacity.com/course/scalable-microservices-with-kubernetes--ud615

https://kubernetes.io/docs/home/

http://omerio.com/2015/12/18/learn-the-kubernetes-key-concepts-in-10-minutes/

https://kubernetes.io/docs/user-guide/kubectl-cheatsheet/

In-depth study:

https://github.com/cncf/curriculum/blob/master/certified_kubernetes_administrator_exam_V0.9.pdf

https://www.edx.org/course/introduction-kubernetes-linuxfoundationx-lfs158x

https://github.com/kelseyhightower/kubernetes-the-hard-way

https://www.digitalocean.com/community/tutorials/how-to-use-journalctl-to-view-and-manipulate-systemd-logs

https://www.digitalocean.com/community/tutorials/systemd-essentials-working-with-services-units-and-the-journal

# Reference Links

Contributing and Troubleshooting:

http://slack.kubernetes.io/

https://stackoverflow.com/questions/tagged/kubernetes

https://github.com/kubernetes/community

Meeting humans:

https://kubernetes.io/community/

https://www.meetup.com/topics/kubernetes/

parispittman@google.com (for starting a Meetup).

https://github.com/kubernetes/community/blob/master/community/office-hours.md

# Thank you

KubeCon2017

The entire Cloud Native Computing Team at Samsung

Special thanks to Aaron Crickenberger, rubber duck par excellence, and Patrick Christopher, bestest mentor and support person.