Kubernetes + Python = ❤

CLOUD NATIVE PRAGUE

2019-09-05

HENNING JACOBS

@try_except_

ZALANDO AT A GLANCE

~ 5.4 billion EUR
revenue 2018

> 250 million
visits per month

> 15.000
employees in Europe

> 79%
of visits via mobile devices

> 26 million
active customers

> 300.000
product choices

~ 2.000
brands

17
countries

zalando

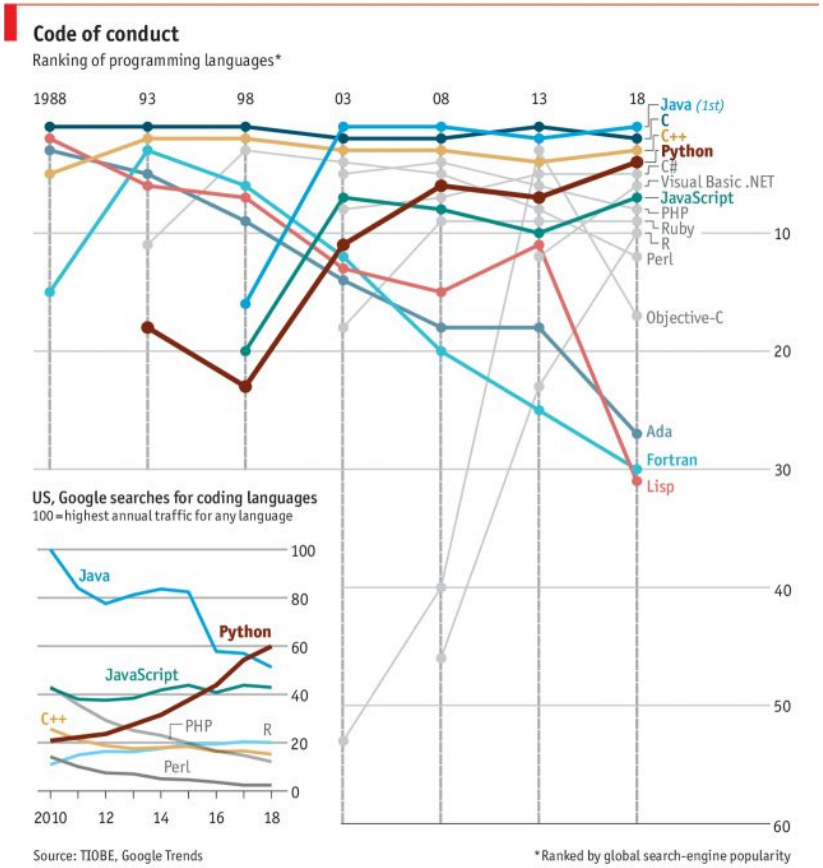
SCALE

396 Accounts

140 Clusters

PYTHON

- Very popular
- Easy to learn
- Huge community



zalando

PYTHON

Michael Gasch
@embano1

Antwort an @try_except_ und @ZalandoTech

thx! asked in several sessions on preferred language from vSphere admin perspective and it is: Python 😀
cc/ @tom_schwaller

Tweet übersetzen

2:59 nachm. · 31. Aug. 2019 · Twitter for iPhone

PYTHON IN ZALANDO

61 (Go) vs 257 (Python) profiles

14k (Go) vs **52k** (Python) internal GitHub hits

Machine Learning / Data Science

patroni

A template for PostgreSQL High Availability with ZooKeeper, etcd, or Consul

kubernetes consul raft postgresql zookeeper etcd haproxy

Python MIT 298 2,703 77 (6 issues need help) 7 Updated 4 hours ago

connexion

Swagger/OpenAPI First framework for Python on top of Flask with automatic endpoint validation & OAuth2 support

python microservices web swagger openapi api-rest flask-extensions

Python 438 2,530 212 (15 issues need help) 51 Updated 4 days ago

RUNNING PYTHON ON KUBERNETES

```python
from aiohttp import web


async def hello(request):
    return web.Response(text="Hello, world")


async def health(request):
    return web.Response(text="OK")


app = web.Application()
app.add_routes([web.get("/", hello)])
app.add_routes([web.get("/health", health)])
# disable SIGTERM handling for disruption-free rolling updates
web.run_app(app, handle_signals=False)
```

RUNNING PYTHON ON KUBERNETES

```
FROM python:3.7-alpine

RUN pip install aiohttp

COPY web.py /

ENTRYPOINT ["python", "web.py"]
```

⇒ ~114 MiB Docker image

DEPLOYMENT MANIFEST

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: aiohttp-helloworld
spec:
  replicas: 1
  selector:
    matchLabels:
      application: aiohttp-helloworld
  template:
    metadata:
      labels:
        application: aiohttp-helloworld
    spec:
      containers:
        - name: mycontainer
          image: hjacobs/aiohttp-helloworld:latest
          imagePullPolicy: IfNotPresent  # For our E2E tests.
          readinessProbe:
            httpGet:
              path: /health
              port: 8080
          resources:
            limits:
              memory: 50Mi
            requests:
              cpu: 5m
              memory: 50Mi
          securityContext:
            readOnlyRootFilesystem: true
            runAsNonRoot: true
            runAsUser: 1000
```

Never without a readinessProbe

Security "stuff"

zalando

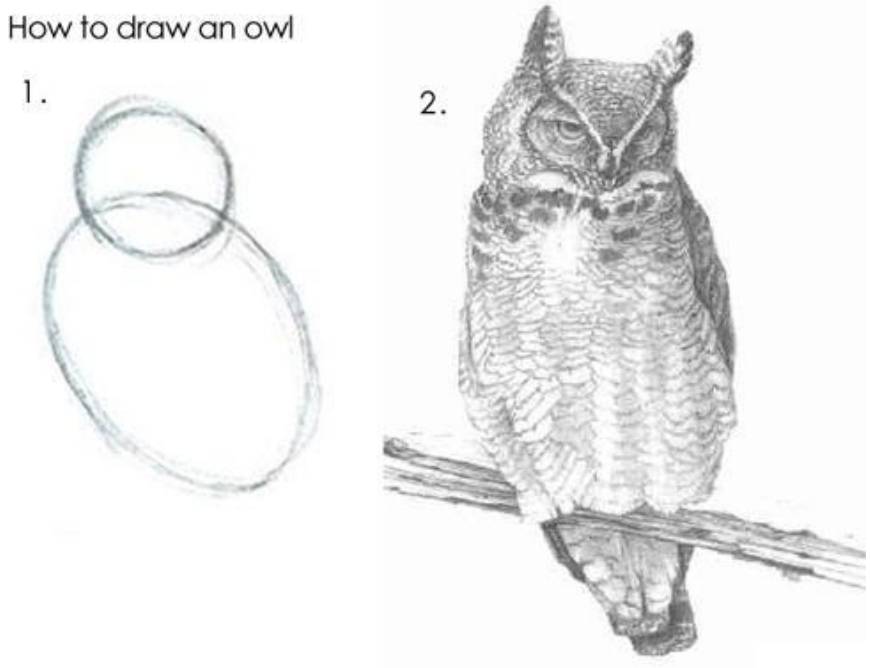
HELLO, WORLD

```
kubectl apply -f deployment.yaml

kubectl port-forward service/aiohttp-helloworld 8080:80

http localhost:8080
```

HTTPie

```
HTTP/1.1 200 OK
Content-Length: 12
Content-Type: text/plain; charset=utf-8
Date: Thu, 05 Sep 2019 09:42:42 GMT
Server: Python/3.7 aiohttp/3.5.4

Hello, world
```

HOW TO USE KUBERNETES



1. run "Hello World"

2. configure the rest of your app

zalando

IS IT FAST ENOUGH?

```
echo 'GET http://localhost:8080' | vegeta attack -rate 200 | vegeta report

Requests      [total, rate]             3300, 200.06
Duration      [total, attack, wait]     16.498s, 16.494, 3.261ms
Latencies     [mean, 50, 95, 99, max]   3.185ms, 3.229ms,
                                        4.182ms, 4.791ms,
                                        21.153ms
Bytes In      [total, mean]             39600, 12.00
Bytes Out     [total, mean]             0, 0.00
Success       [ratio]                   100.00%
Status Codes  [code:count]              200:3300
```

local run on my laptop, single Pod in kind cluster

RUNNING PYTHON ON KUBERNETES

```python
from aiohttp import web


async def hello(request):
    return web.Response(text="Hello, world")


async def health(request):
    return web.Response(text="OK")


app = web.Application()
app.add_routes([web.get("/", hello)])
app.add_routes([web.get("/health", health)])
# disable SIGTERM handling for disruption-free rolling updates
web.run_app(app, handle_signals=False)
```

"GRACEFUL" TERMINATION IN KUBERNETES

1. kubectl delete pod
2. **SIGTERM** + remove endpoints
3. grace period (default: 30s)
4. **SIGKILL**

kube-proxy/ iptables, Load Balancer update

⇒ **Going down on SIGTERM leads to request errors** 

TESTING THE ROLLING UPDATE

tests/e2e/test_update.py::test_rolling_update_no_signal_handling

Elapsed: 41.60 seconds
Successes: 2056
Errors: 0
Requests/s: 49.42 rps

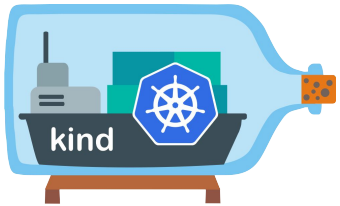
tests/e2e/test_update.py::test_rolling_update_with_signal_handling

Elapsed: 15.16 seconds
Successes: 816
Errors: **2**
Requests/s: 53.95 rps

ALTERNATIVE: THE PRESTOP TRICK

```yaml
containers:
  - name: mycontainer

    ...

    # alternative approach to ensure safe rolling updates
    # add additional sleep of 20 seconds
    # the hook will be executed before sending SIGTERM
    lifecycle:
      preStop:
        exec:
          command: ["sleep","20"]
```

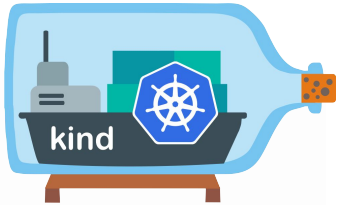
TESTING WITH PYTEST AND KIND




```python
import requests


def test_web_hello_world(kind_cluster):
    kind_cluster.load_docker_image("hjacobs/aiohttp-helloworld:latest")
    kind_cluster.kubectl("apply", "-f", "deployment.yaml")

    # wait for rolling update
    kind_cluster.kubectl("rollout", "status", "deployment/aiohttp-helloworld")

    # now check whether our app returns the expected text
    with kind_cluster.port_forward("service/aiohttp-helloworld", 80) as port:
        response = requests.get(f"http://localhost:{port}/")
        response.raise_for_status()
        assert response.text == "Hello, world"
```

```
docker build -t hjacobs/aiohttp-helloworld:latest .
Sending build context to Docker daemon  94.88MB
Step 1/4 : FROM python:3.7-alpine
 ---> 39fb80313465
Step 2/4 : RUN pip install aiohttp
 ---> Using cache
 ---> 75e094c4447b
Step 3/4 : COPY web.py /
 ---> Using cache
 ---> dc25bceb0482
Step 4/4 : ENTRYPOINT ["python", "web.py"]
 ---> Using cache
 ---> f6160a68a628
Successfully built f6160a68a628
Successfully tagged hjacobs/aiohttp-helloworld:latest
poetry install
Installing dependencies from lock file

Nothing to install or update

poetry run pytest -r=a \
               --log-cli-level info \
               --log-cli-format '%(asctime)s %(levelname)s %(message)s' \
               --cluster-name=aiohttp-helloworld
============================================================================================================================ test session starts =============================
platform linux -- Python 3.7.3, pytest-5.1.2, py-1.8.0, pluggy-0.12.0
rootdir: /home/hjacobs/workspace/kubernetes-and-python/01-running-python-on-kubernetes/aiohtttp-helloworld
plugins: kind-19.9.0
collected 1 item

tests/test_web.py::test_web_hello_world
------------------------------------------------------------------------------------------------------------------------------ live log setup ---------------------------------
21:10:19 INFO Creating cluster aiohttp-helloworld..
------------------------------------------------------------------------------------------------------------------------------ live log call ----------------------------------
21:10:43 INFO Loading Docker image hjacobs/aiohttp-helloworld:latest in cluster (usually ~5s)..
PASSED
------------------------------------------------------------------------------------------------------------------------------ live log teardown ------------------------------
21:11:26 INFO Deleting cluster aiohttp-helloworld..


============================================================================================================================ 1 passed in 67.97s (0:01:07) ============================
```

TESTING WITH PYTEST AND KIND



pytest-kind

build passing | pypi v19.9.1 | python 3.7 | license GPL-3.0 | calver YY.MM.MICRO

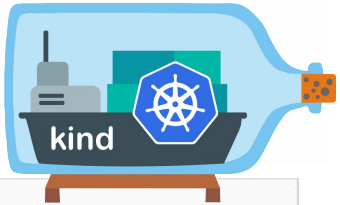
Test your Python Kubernetes app/operator end-to-end with kind and pytest.

`pytest-kind` is a plugin for pytest which provides the `kind_cluster` fixture. The fixture will install kind 0.5.1, create a Kubernetes 1.15 cluster, and provide convenience functionality such as port forwarding.

https://pypi.org/project/pytest-kind/

USING KIND CLUSTER DIRECTLY

pytest-kind




```python
from pytest_kind import KindCluster

cluster = KindCluster("myclustername")
cluster.create()
cluster.kubectl("apply", "-f", "..")
# ...
cluster.delete()
```

https://pypi.org/project/pytest-kind/

ACCESSING KUBERNETES FROM PYTHON

zalando

ACCESSING KUBERNETES FROM PYTHON

- Official Kubernetes client - github.com/kubernetes-client/python
- Pykube-ng - github.com/hjacobs/pykube

```
du -csh /../../lib/python3.7/site-packages/kubernetes/
23M    total

du -csh /../../lib/python3.7/site-packages/pykube/
160K   total
```

LISTING CONTAINER IMAGES

```python
from kubernetes import client, config, watch

config.load_kube_config()
v1 = client.CoreV1Api()
ret = v1.list_namespaced_pod(args.namespace)
for pod in ret.items:
    images = [c.image for c in pod.spec.containers]
    print(pod.metadata.name, ", ".join(images))
```

```python
from pykube import HTTPClient, KubeConfig, Pod

api = HTTPClient(KubeConfig.from_file())

for pod in Pod.objects(api).filter(namespace=args.namespace):
    images = [c["image"] for c in pod.obj["spec"]["containers"]]
    print(pod.name, ", ".join(images))
```

OFFICIAL CLIENT VS PYKUBE-NG

Kubernetes Python Client

- "Official"
- Generated (Swagger Codegen)
- Less "Pythonic"
- urllib3
- Supports many auth methods
- Heavy (23 MiB)
- Python 2.7 and 3

pykube-ng

- Kel project 2015
- Dynamic
- More "Pythonic"
- requests
- Limited auth methods
- Light (160 KiB)
- Only Python 3

zalando

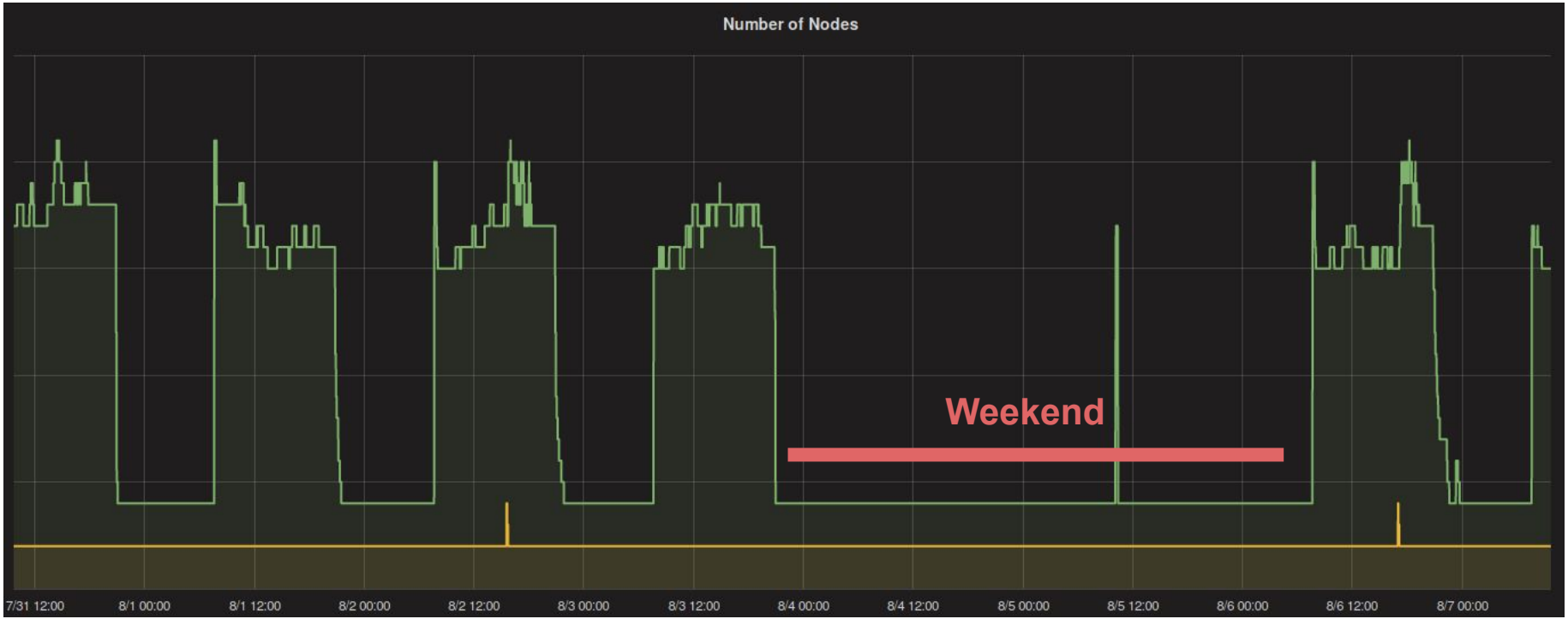
INTERACTIVE CONSOLE

```
$ pip3 install pykube-ng

$ python3 -m pykube
Pykube v19.9.1, loaded "/home/hjacobs/.kube/config" with context "mycluster".

    Example commands:
      [d.name for d in Deployment.objects(api)]             # get names of deployments in default namespace
      list(DaemonSet.objects(api, namespace='kube-system'))  # list daemonsets in "kube-system"
      Pod.objects(api).get_by_name('mypod').labels          # labels of pod "mypod"

    Use Ctrl-D to exit
>>>
```

WRITING A SIMPLE CONTROLLER

```
import datetime, os, pykube, time

while True:
    # loads in-cluster auth or local ~/.kube/config for testing
    config = pykube.KubeConfig.from_env()
    api = pykube.HTTPClient(config)

    # no timezone handling!
    weekday = datetime.datetime.today().isoweekday()
    is_weekend = weekday in (6, 7)

    for deploy in pykube.Deployment.objects(api, namespace=pykube.all):
        if "scale-down-on-weekend" in deploy.annotations and is_weekend:
            print(f"Updating deployment {deploy.namespace}/{deploy.name}..")
            deploy.replicas = 0
            deploy.update()
    time.sleep(15)
```

DOWNSCALING DURING OFF-HOURS



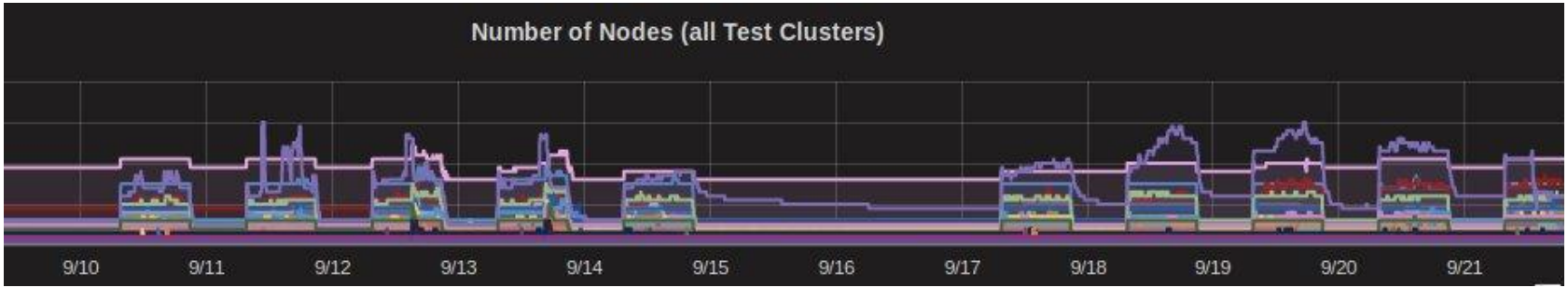
github.com/hjacobs/kube-downscaler

zalando

DOWNSCALING DURING OFF-HOURS

```
DEFAULT_UPTIME="Mon-Fri 07:30-20:30 CET"
```

```
annotations:
    downscaler/exclude: "true"
```

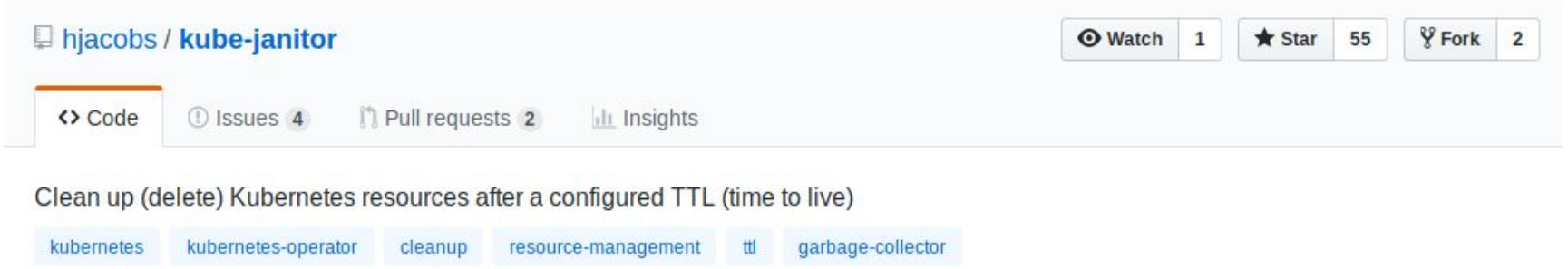


github.com/hjacobs/kube-downscaler

HOUSEKEEPING

- Delete prototypes after X days
- Clean up temporary deployments
- Remove resources without owner

KUBERNETES JANITOR



- **TTL** and **expiry date** annotations, e.g.
  - set time-to-live for your test deployment
- **Custom rules**, e.g.
  - delete everything without "app" label after 7 days

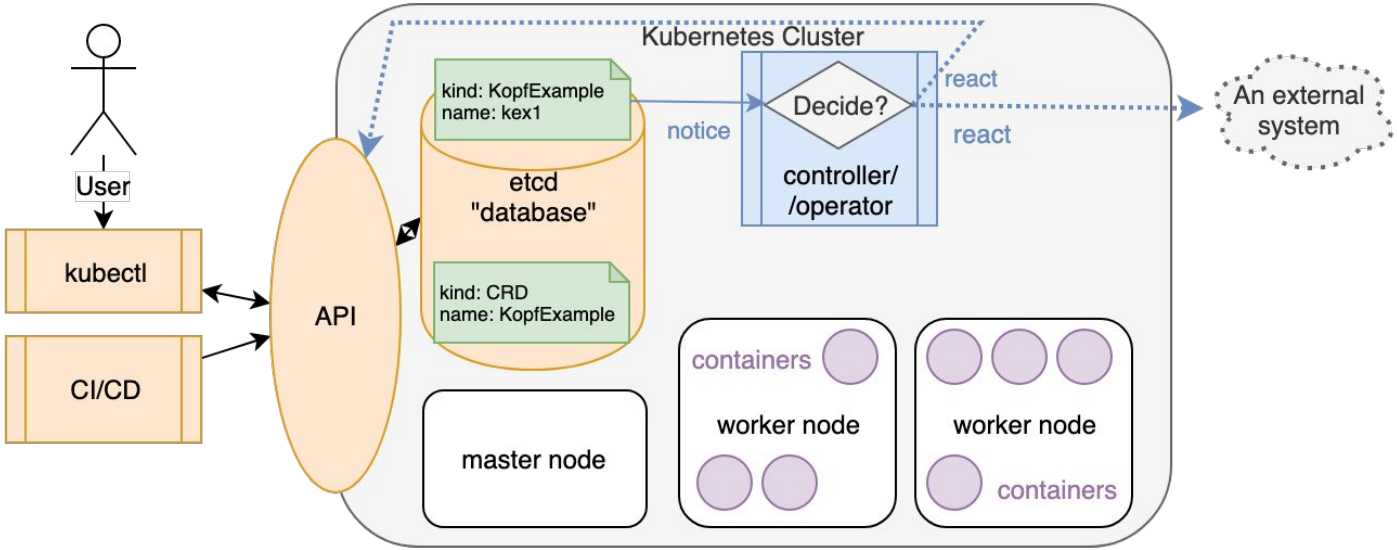
github.com/hjacobs/kube-janitor

JANITOR TTL ANNOTATION

```
# let's try out nginx, but only for 1 hour
kubectl run nginx --image=nginx
kubectl annotate deploy nginx janitor/ttl=1h
```

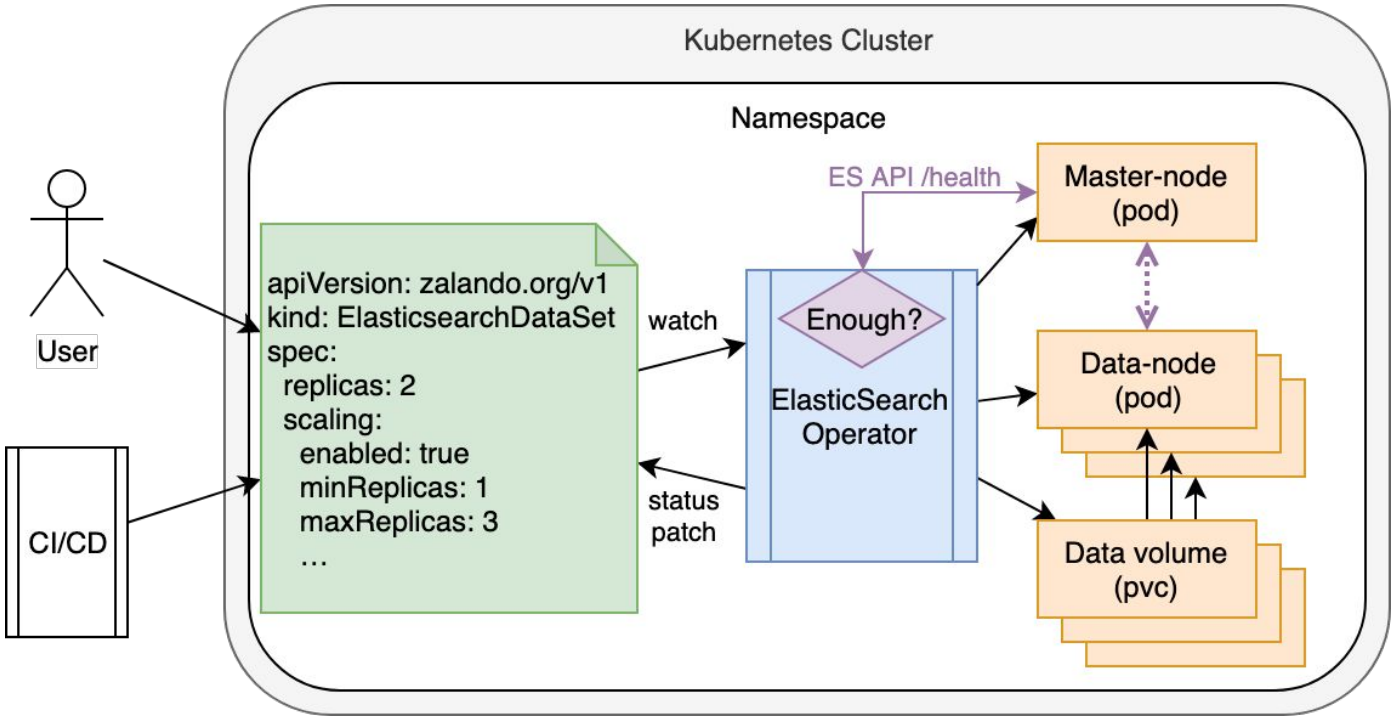

github.com/hjacobs/kube-janitor

EXTENDING KUBERNETES WITH OPERATORS



zalando

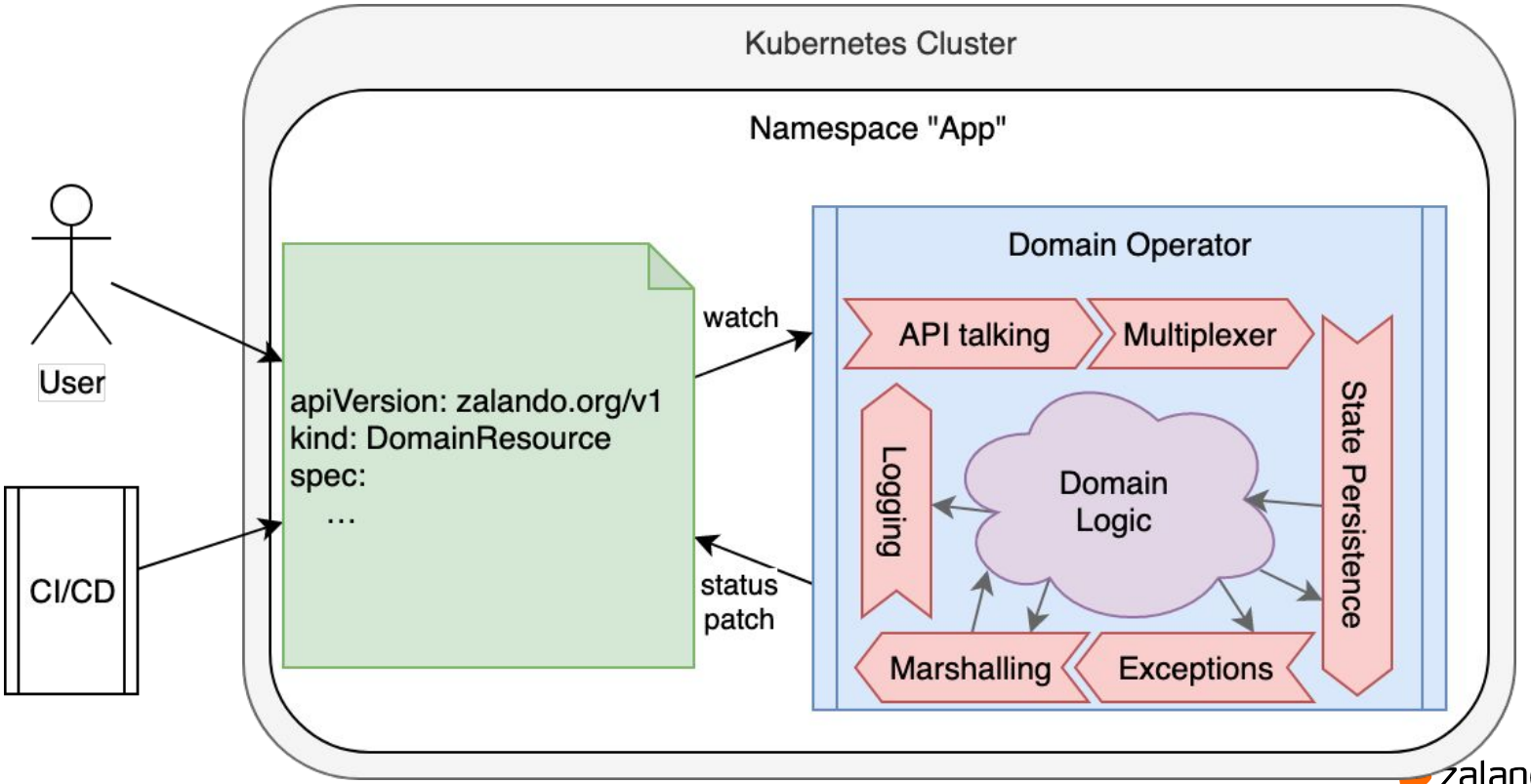
ELASTICSEARCH OPERATOR



github.com/zalando-incubator/es-operator

zalando

WRITING AN OPERATOR



zalando

MAKE A FRAMEWORK!

Kubernetes Operator Pythonic Framework (Kopf)

Kopf —Kubernetes Operator Pythonic Framework— is a framework and a library to make Kubernetes operators development easier, just in few lines of Python code.

github.com/zalando-incubator/kopf

KOPF: SIMPLE SPY-HANDLERS

React on events from K8s API.
Raw payload, no interpretation. Fire-and-forget, ignore errors.

```python
import kopf


@kopf.on.event('zalando.org', 'v1', 'kopfexamples')
def event_fn_with_error(**kwargs):
    raise Exception("Oops!")


@kopf.on.event('zalando.org', 'v1', 'kopfexamples')
def normal_event_fn(event, **kwargs):
    print(f"Event received: {event!r}")

```

KOPF: CONVENIENT CAUSE AND DIFF HANDLERS

```python
import kopf

@kopf.on.create('zalando.org', 'v1', 'kopfexamples')
def create_fn_1(spec, **kwargs):
    print(f'CREATED 1st: field={spec.field}')

@kopf.on.create('zalando.org', 'v1', 'kopfexamples')
def create_fn_2(meta, **kwargs):
    print(f'CREATED 2nd: name={meta["name"]}')

@kopf.on.update('zalando.org', 'v1', 'kopfexamples')
def update_fn(old, new, diff, **kwargs):
    print(f'UPDATED: diff={diff}')

@kopf.on.delete('zalando.org', 'v1', 'kopfexamples')
def delete_fn_1(**kwargs):
    print('DELETED')

@kopf.on.field('zalando.org', 'v1', 'kopfexamples', field='spec.field')
def field_fn(old, new, **kwargs):
    print(f'FIELD CHANGED: {old} -> {new}')
```

RUNNING FROM DEVELOPMENT ENVIRONMENT

```python
import kopf


@kopf.on.create('zalando.org', 'v1', 'kopfexamples')
def create_fn(spec, **kwargs):
    print(f"And here we are! Creating: {spec}")
    return {'message': 'hello world'}  # will be the new status
```

```yaml
# A demo custom resource for
apiVersion: zalando.org/v1
kind: KopfExample
metadata:
  name: kopf-example-1
  labels:
    somelabel: somevalue
spec:
  duration: 1m
  field: value
  items:
  - item1
  - item2
```

```
$ kopf run scripts.py [--verbose]
```

```
And here we are! Creating: {'duration': '1m', 'field': 'value',
'items': ['item1', 'item2']}

[2019-02-25 14:06:54,742] kopf.reactor.handlin [INFO    ]
[asf-preprocessing/kopf-example-1] Handler create_fn succeeded.

[2019-02-25 14:06:54,856] kopf.reactor.handlin [INFO    ]
[asf-preprocessing/kopf-example-1] All handlers succeeded for
creation.
```

```
$ kubectl apply -f ../obj.yaml
$ kubectl describe -f ../obj.yaml
```

```
Name:         kopf-example-1
...
Status:
  create_fn:
    Message:  hello world
Events:
  Type    Reason   Age   From  Message
  ----    ------   ----  ----  -------
  Normal  Success  81s   kopf  Handler create_fn succeeded.
```

"HELLO WORLD" EXAMPLE: CRD

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: helloworlds.example.org
spec:
  scope: Namespaced
  group: example.org
  versions:
    - name: v1
      served: true
      storage: true
  names:
    kind: HelloWorld
    plural: helloworlds
    singular: helloworld
    shortNames:
      - hw
```

github.com/zalando-incubator/kopf

8 LINES OF PYTHON

```python
import kopf


@kopf.on.create("example.org", "v1", "helloworlds")
def on_create(spec, **kwargs):
    print(f"Create handler is called with spec: {spec}")
    return {"message": f"Hello {spec['name']}!"}


@kopf.on.update("example.org", "v1", "helloworlds")
def on_update(body, **kwargs):
    print(f"Update handler is called with body: {body}")
```


github.com/zalando-incubator/kopf

4 LINES OF DOCKER

```
FROM python:3.7-alpine

RUN pip install kopf

COPY handlers.py /

ENTRYPOINT ["kopf", "run", "handlers.py"]
```


github.com/zalando-incubator/kopf

CRD OBJECT

```yaml
apiVersion: example.org/v1
kind: HelloWorld
metadata:
    name: test-1
spec:
    name: "Prague"
```

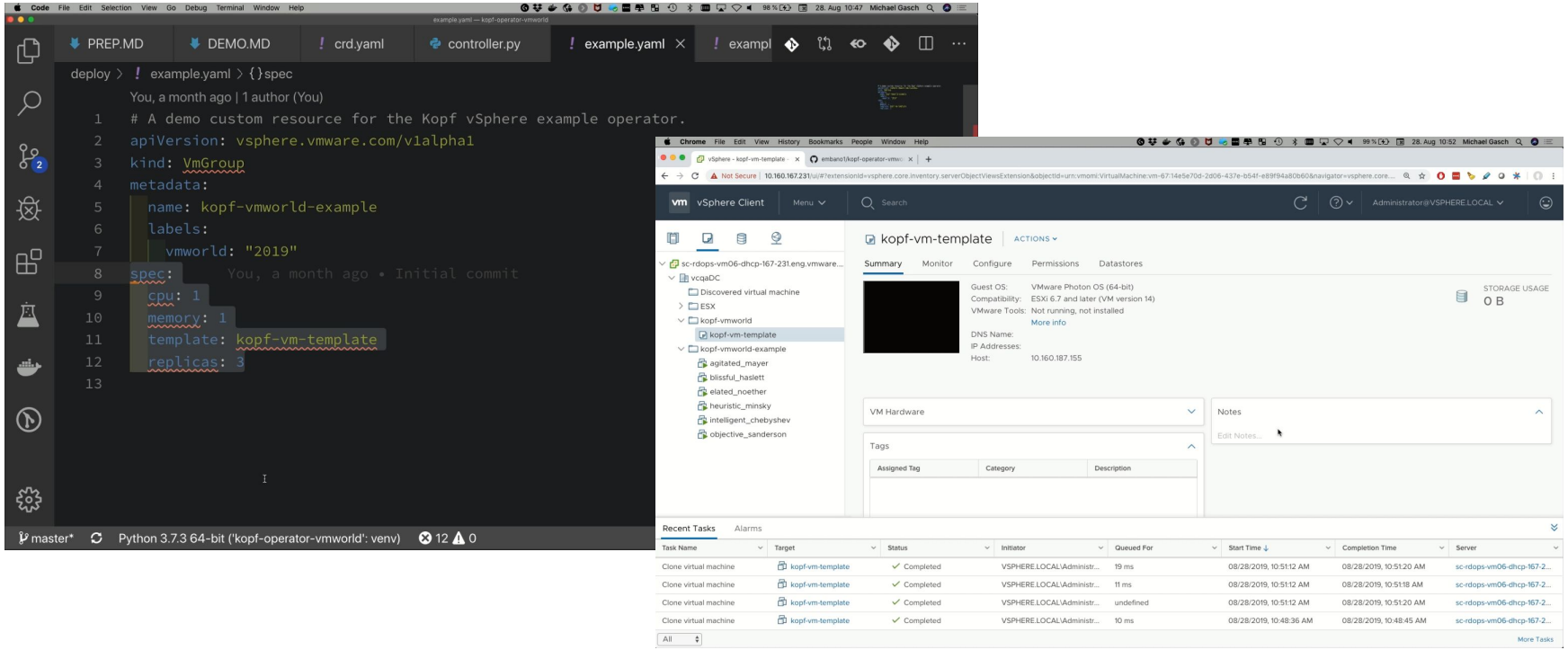

github.com/zalando-incubator/kopf

STATUS: "HELLO PRAGUE!"

```
apiVersion: example.org/v1
kind: HelloWorld
metadata:
  annotations:
    kopf.zalando.org/last-handled-configuration: '{"spec": {"name": "Prague"}}'
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"example.org/v1","kind":"HelloWorld","metadata":{"annotations":{},"name":"test-1","namespace":"default"},'
  creationTimestamp: 2019-09-04T19:32:59Z
  generation: 2
  name: test-1
  namespace: default
  resourceVersion: "763"
  selfLink: /apis/example.org/v1/namespaces/default/helloworlds/test-1
  uid: 2c313666-118c-4a25-8c58-fbcea6ffd686
spec:
  name: Prague
status:
  kopf: {}
  on_create:
    message: Hello Prague!
```


github.com/zalando-incubator/kopf

VMWORLD 2019 SESSION WITH KOPF



github.com/embano1/kopf-operator-vmworld

zalando

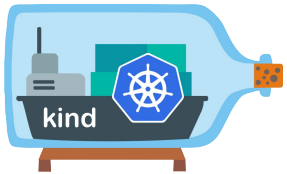
KOPF: FEATURES

- Custom & built-in resources supported (CRDs, Pods, Services, etc)
- Agnostic to API clients: kubernetes-client, pykube-ng, raw HTTP, etc
- Immediate reaction to changes and events
- Predefined behavioural patterns:
  - Simple spy-handlers for event watching
  - Advanced cause & diff detection for actual change tracking
  - Retry-until-success approach to handlers
- Operator resilience:
  - Restores its state on restarts
- Operator testing toolkit (minimally sufficient)

github.com/zalando-incubator/kopf

TESTING WITH KIND

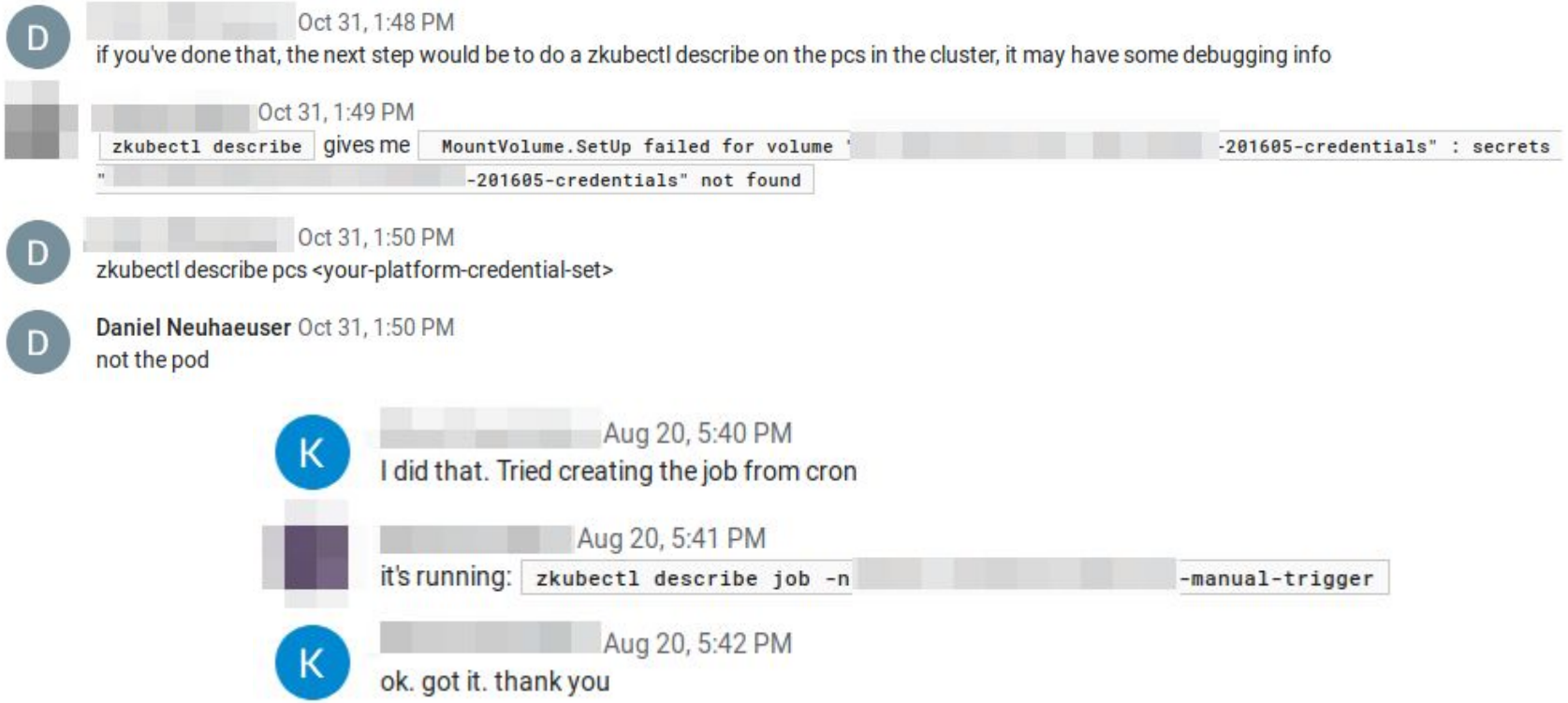
```python
from pykube.objects import NamespacedAPIObject


class HelloWorld(NamespacedAPIObject):
    version = "example.org/v1"
    kind = "HelloWorld"
    endpoint = "helloworlds"


def test_web_hello_world(kind_cluster):
    kind_cluster.load_docker_image("kopf-example")
    kind_cluster.kubectl("apply", "-f", "deploy/")

    kind_cluster.kubectl("rollout", "status", "deployment/kopfexample-operator")

    kind_cluster.kubectl("apply", "-f", "example.yaml")

    for i in range(10):
        obj = HelloWorld.objects(kind_cluster.api).get(name="test-1")
        if "status" in obj.obj:
            break
        time.sleep(2)

    assert obj.obj["status"]["on_create"]["message"] == "Hello Prague!"
```



Kubernetes Web View

"kubectl for the web"

MOTIVATION: CHAT PING-PONG



zalando

MOTIVATION: PENDING PODS INCIDENT

Title of Incident: Multiple Kubernetes test clusters have “Pending” pods, deployments are stuck
Ticket(s):
Document Owner:
Contributor:
Team(s):
Folder:

Short summary

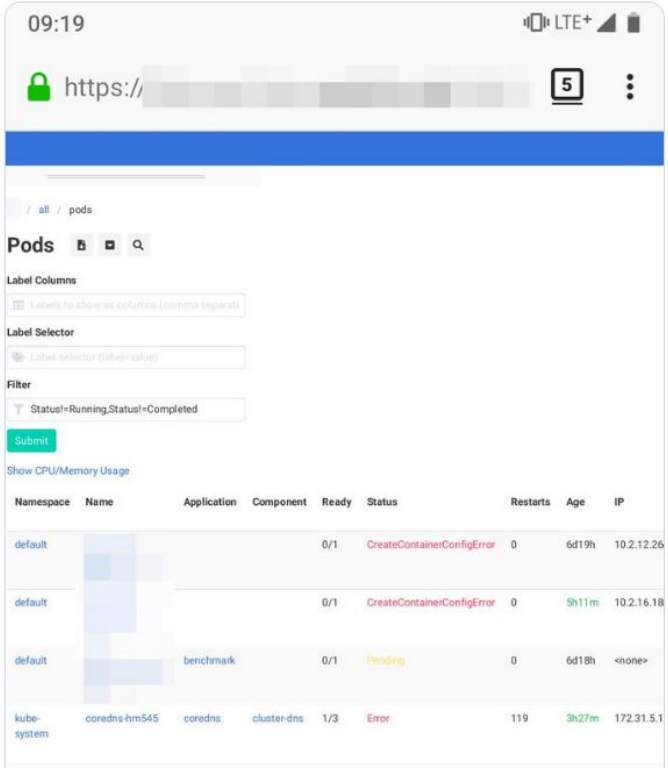
Due to a mistake in copying the custom Ubuntu AMI the test clusters were configured to run a PR branch of the AMI which was deregistered. As a result test clusters could not create new instances to replace existing spot instances whose lease had expired leading to almost complete unavailability of test clusters.

⇒ I want to see all "Pending" Pods across all clusters!

Henning Jacobs
@try_except_

How many incidents happen when you are on the way 🚋 to work? While the mobile UX of kube-web-view is not perfect yet, troubleshooting problematic #Kubernetes pods is already helpful 😊

Tweet übersetzen



zalando

SOME USE CASES

Support interactions with colleagues

- share links
- avoid human mistakes (wrong cluster login on CLI, "-n" param, ..)
- deep link to resource YAML

Incident response and investigation

- find a resource by name across clusters to help troubleshoot
- find pending pods across all clusters
- help identify if a problem is only in one cluster or multiple ones

KUBERNETES WEB UIS IN 2019

Posted: 2019-08-12 22:24 | More posts about kubernetes

This post takes a look at different open source Kubernetes web UIs, my requirements, and why I created Kubernetes Web View to help with support and troubleshooting across multiple clusters.

Use Cases

At Zalando, we have a large number of Kubernetes users (900+) and clusters (100+). There are two main use cases where I would wish for some web tool support: 1) support interactions with colleagues 2) incident response and investigation.

Support

I observed that support interactions in chat often look like:

- A: Help our service XYZ is not reachable
- B: What do you see when you do `kubectl describe ingress ...`?

Or similar for CRDs:

- A: I am facing an issue with identity service, ...
- B: What is the output of `kubectl describe platformcredentialsset ...`?

These interactions follow a pattern of using the `kubectl` command line to investigate and pinpoint a problem. This requires context switches on both sides: participants have to switch between terminal and web chat, plus they don't see the same situation.

I would wish for a Kubernetes web frontend to help with this:

- users **can share links** and see the same situation
- **avoid human mistakes** during support: login to wrong cluster on the command line, typos in CLI commands, etc
- **allow crafting custom views** to send to colleagues, i.e. add label columns, show multiple resource types on one page
- ideally the web tool should allow **deep links into specific YAML sections** (e.g. to point to a wrong spec causing the issues)

Incident Response & Investigation

Incident response for infrastructure requires gaining situational awareness, the ability to assess impact, and finding patterns across clusters. Some real-life example situations:

srcco.de/posts/kubernetes-web-uis-in-2019.html

zalando

Search

Search Text pipeline-id=l-2pqt9gs2ia81aef95hwcbtrdjv Search!

Resource Types ☐ CronJob ☐ DaemonSet ☑ Deployment ☑ Ingress ☐ Namespace ☐ Node ☐ Pod ☐ ReplicaSet ☑ Service ☐ StatefulSet ✕ unselect all

kube-web-view (Deployment)

[redacted]aces/default/deployments/kube-web-view
Created: 2019-07-26 12:34:50
application: kube-web-view | deployment-id: d-293jpa847d74p32js5bxo1modc | pipeline-id: l-2pqt9gs2ia81aef95hwcbtrdjv

kube-web-view (Ingress)

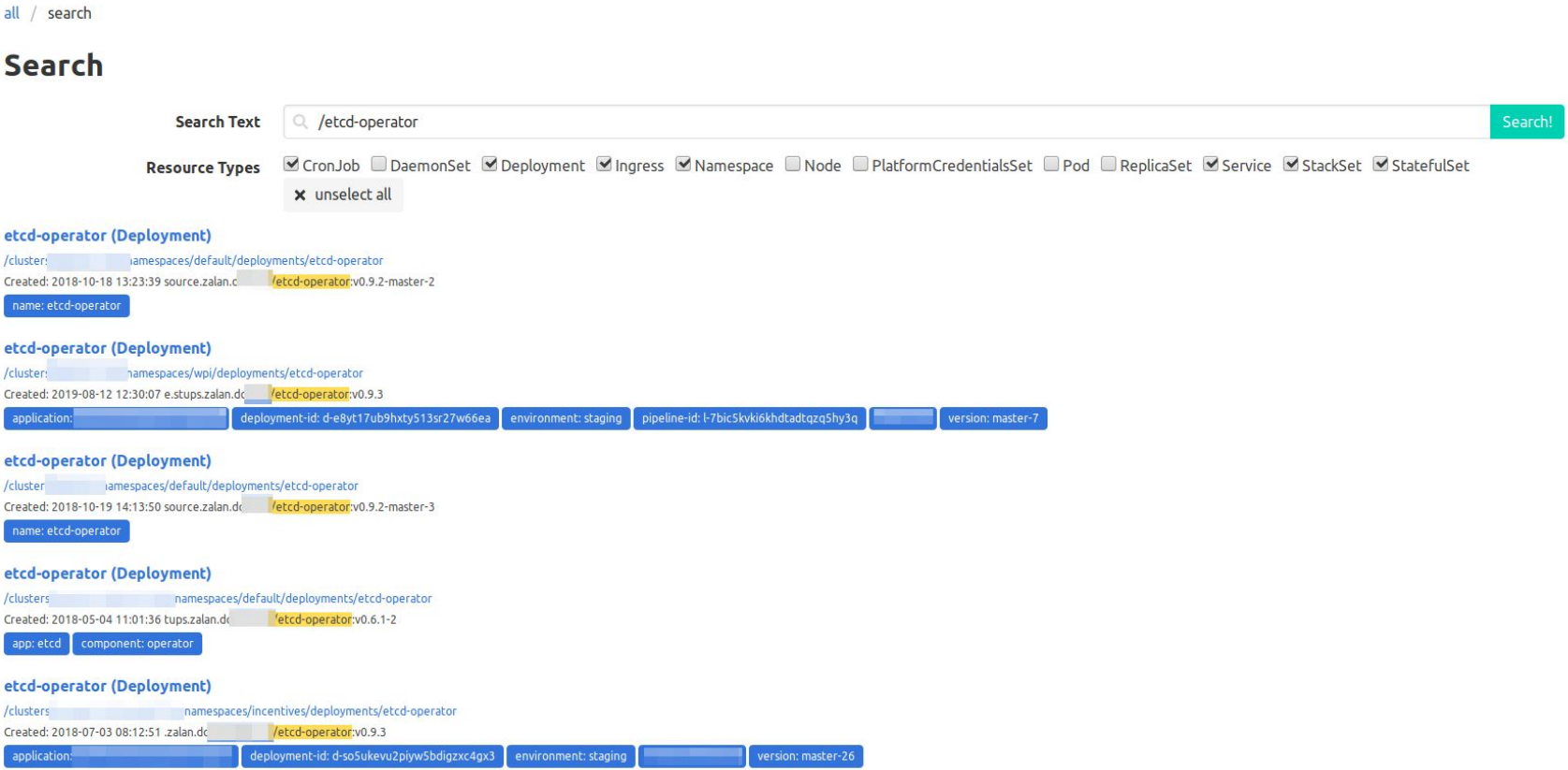
[redacted]aces/default/ingresses/kube-web-view
Created: 2019-07-26 12:34:51
deployment-id: d-293jpa847d74p32js5bxo1modc | pipeline-id: l-2pqt9gs2ia81aef95hwcbtrdjv

kube-web-view (Service)

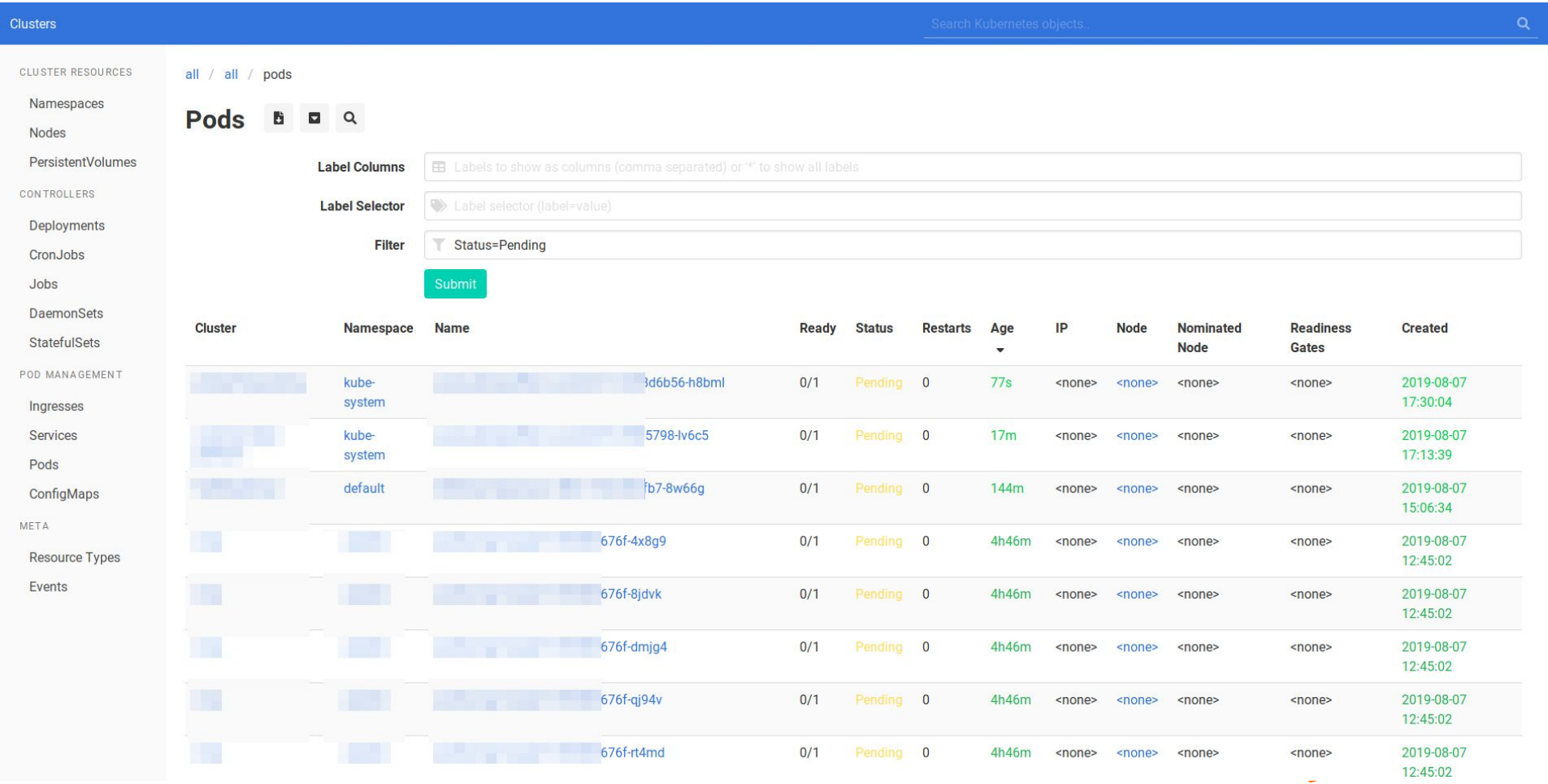
[redacted]paces/default/services/kube-web-view
Created: 2019-07-24 06:38:20
deployment-id: d-293jpa847d74p32js5bxo1modc | pipeline-id: l-2pqt9gs2ia81aef95hwcbtrdjv

3 results found. Searched 3 resource types in 146 clusters in 2.090 seconds.

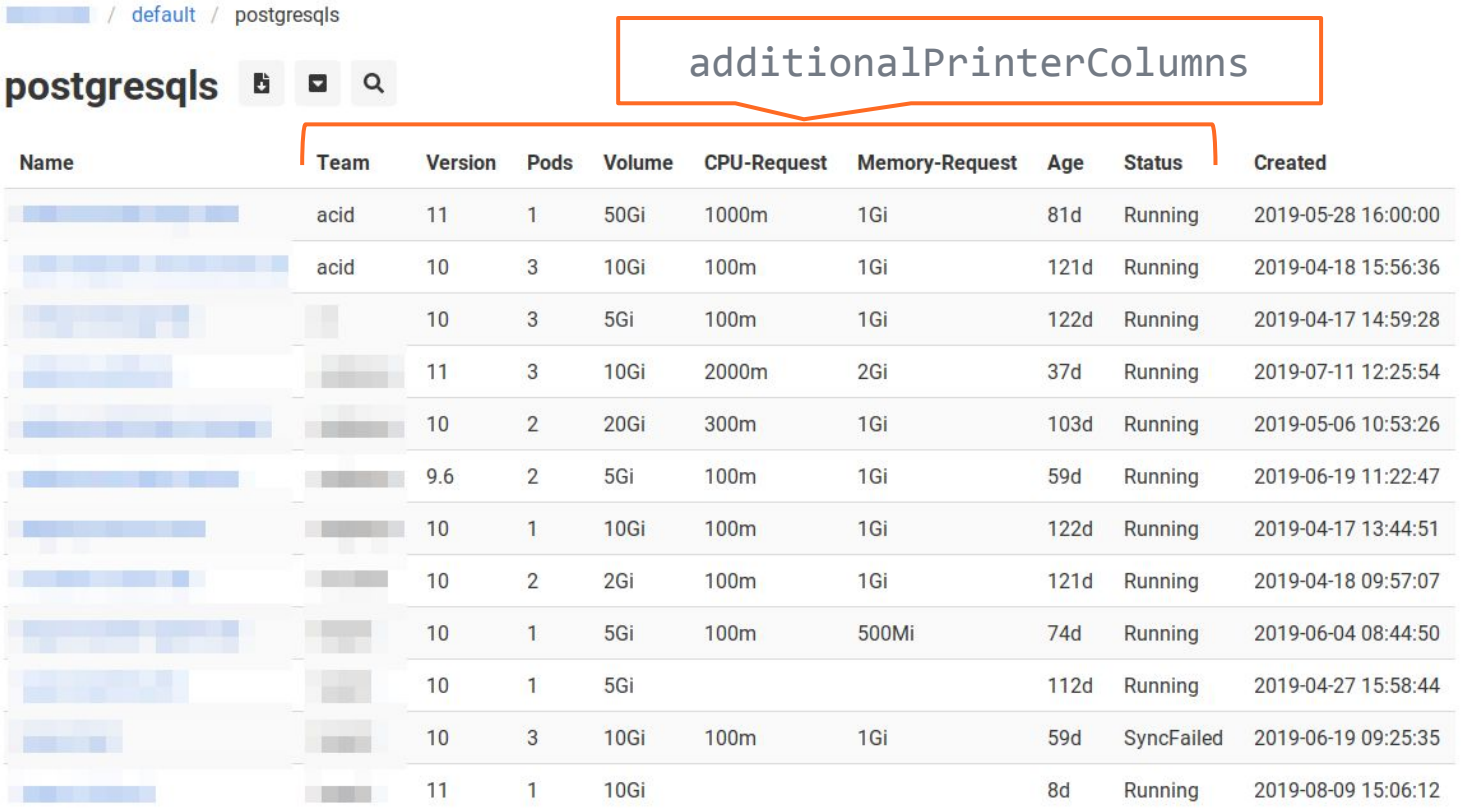
SEARCHING FOR DOCKER IMAGE



zalando


Clusters
Search Kubernetes objects..
CLUSTER RESOURCES
Namespaces
Nodes
PersistentVolumes
CONTROLLERS
Deployments
CronJobs
Jobs
DaemonSets
StatefulSets
POD MANAGEMENT
Ingresses
Services
Pods
ConfigMaps
META
Resource Types
Events
all / all / pods
Pods
Label Columns
Labels to show as columns (comma separated) or '*' to show all labels
Label Selector
Label selector (label=value)
Filter
Status=Pending
Submit
Cluster
Namespace
Name
Ready
Status
Restarts
Age
IP
Node
Nominated Node
Readiness Gates
Created
kube-system
3d6b56-h8bml
0/1
Pending
0
77s
<none>
<none>
<none>
<none>
2019-08-07 17:30:04
kube-system
5798-lv6c5
0/1
Pending
0
17m
<none>
<none>
<none>
<none>
2019-08-07 17:13:39
default
fb7-8w66g
0/1
Pending
0
144m
<none>
<none>
<none>
<none>
2019-08-07 15:06:34
676f-4x8g9
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-8jdvk
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-dmjg4
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-qj94v
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02
676f-rt4md
0/1
Pending
0
4h46m
<none>
<none>
<none>
<none>
2019-08-07 12:45:02

CRDS

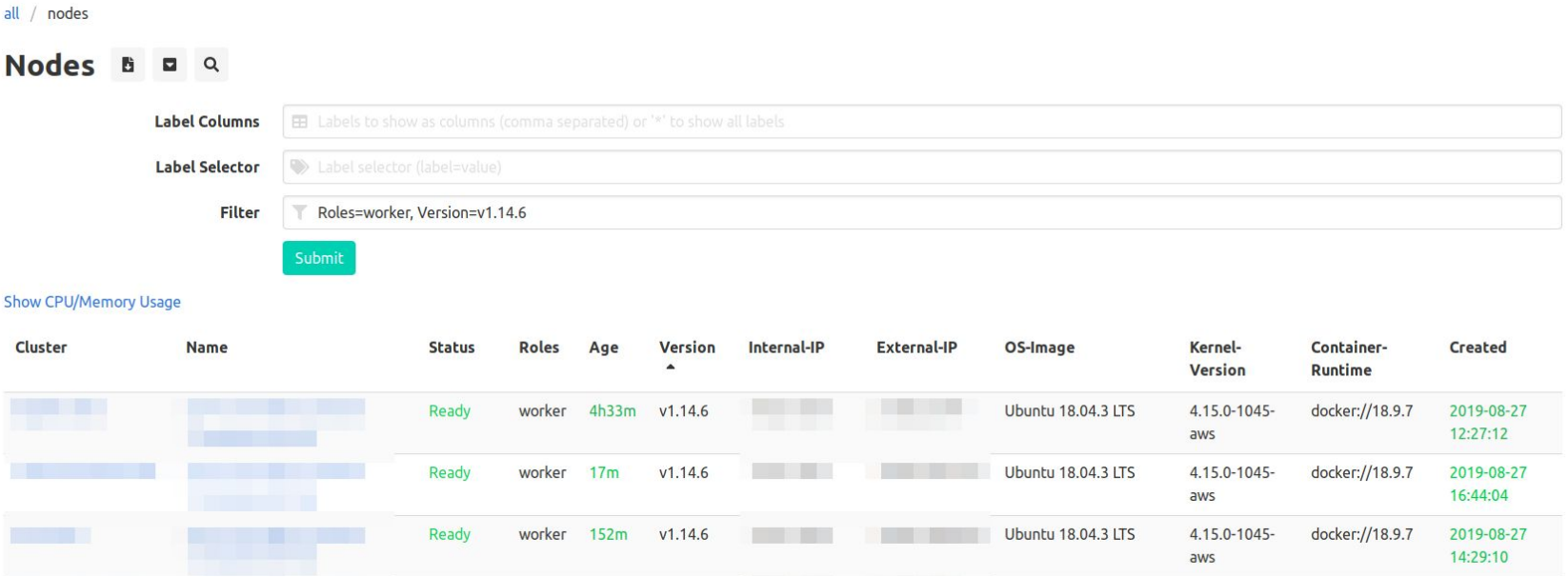


Name	Team	Version	Pods	Volume	CPU-Request	Memory-Request	Age	Status	Created
	acid	11	1	50Gi	1000m	1Gi	81d	Running	2019-05-28 16:00:00
	acid	10	3	10Gi	100m	1Gi	121d	Running	2019-04-18 15:56:36
		10	3	5Gi	100m	1Gi	122d	Running	2019-04-17 14:59:28
		11	3	10Gi	2000m	2Gi	37d	Running	2019-07-11 12:25:54
		10	2	20Gi	300m	1Gi	103d	Running	2019-05-06 10:53:26
		9.6	2	5Gi	100m	1Gi	59d	Running	2019-06-19 11:22:47
		10	1	10Gi	100m	1Gi	122d	Running	2019-04-17 13:44:51
		10	2	2Gi	100m	1Gi	121d	Running	2019-04-18 09:57:07
		10	1	5Gi	100m	500Mi	74d	Running	2019-06-04 08:44:50
		10	1	5Gi			112d	Running	2019-04-27 15:58:44
		10	3	10Gi	100m	1Gi	59d	SyncFailed	2019-06-19 09:25:35
		11	1	10Gi			8d	Running	2019-08-09 15:06:12

zalando

UPGRADE TO KUBERNETES 1.14

"Found 1223 rows for 1 resource type in 148 clusters in 3.301 seconds."



Cluster	Name	Status	Roles	Age	Version	Internal-IP	External-IP	OS-Image	Kernel-Version	Container-Runtime	Created
[redacted]	[redacted]	Ready	worker	4h33m	v1.14.6	[redacted]	[redacted]	Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 12:27:12
[redacted]	[redacted]	Ready	worker	17m	v1.14.6	[redacted]	[redacted]	Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 16:44:04
[redacted]	[redacted]	Ready	worker	152m	v1.14.6	[redacted]	[redacted]	Ubuntu 18.04.3 LTS	4.15.0-1045-aws	docker://18.9.7	2019-08-27 14:29:10

zalando

RESOURCE YAML VIEW

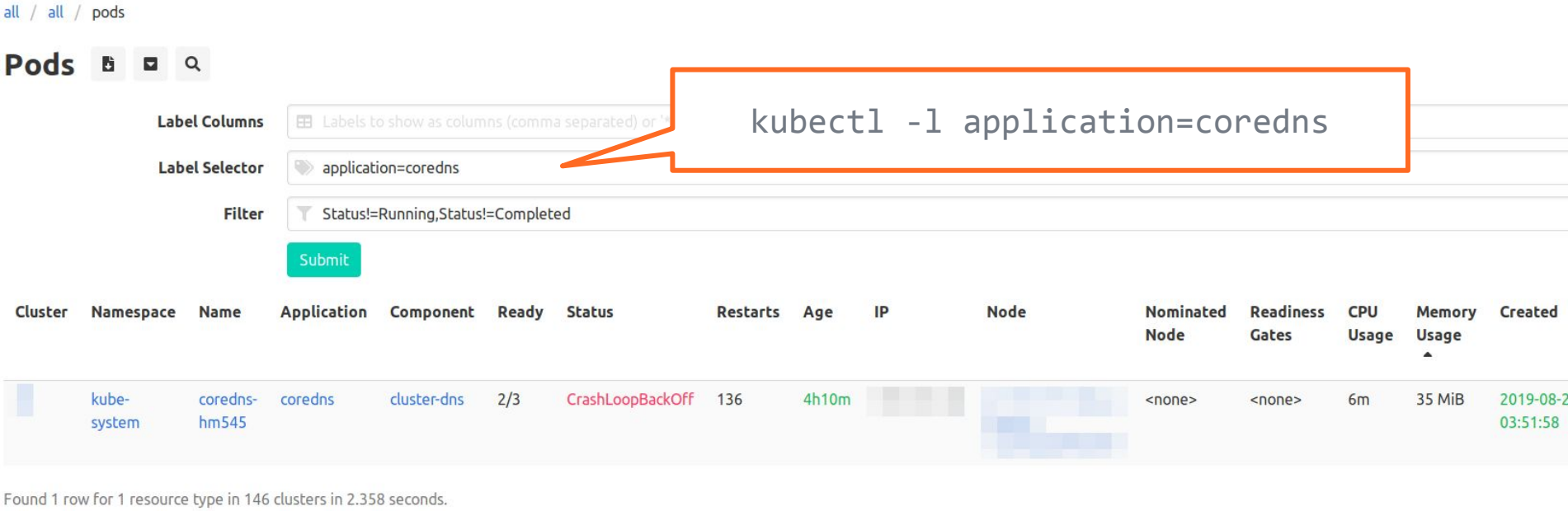
teapot / e2e-tests-zalando-kube-admission-controller-cvbcz / pods / deployment-info-test-fabc1348-b2ae-11e9-a3a6-784f4384f2b4-mj9z5

deployment-info-test-fabc1348-b2ae-11e9-a3a6-784f4384f2b4-mj9z5

Default YAML Logs

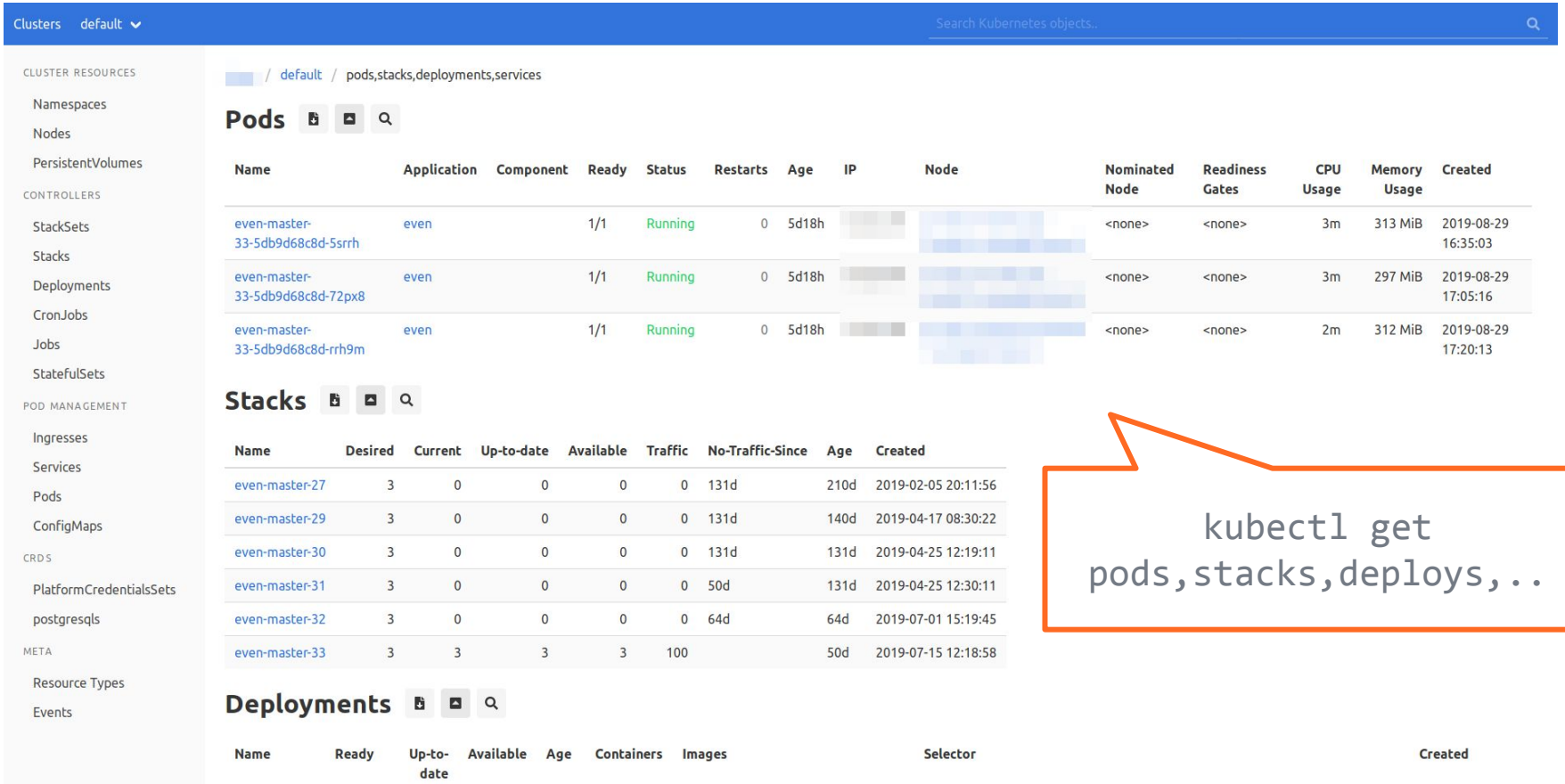
```yaml
apiVersion: v1
kind: Pod
metadata:
  annotations:
    kubernetes.io/psp: e2e-test-privileged-psp
  creationTimestamp: '2019-08-15T08:40:17Z'
  generateName: deployment-info-test-fabc1348-b2ae-11e9-a3a6-784f4384f2b4-dd77cb479-
  labels:
    application: e2e-test-application
    pod-template-hash: dd77cb479
  name: deployment-info-test-fabc1348-b2ae-11e9-a3a6-784f4384f2b4-mj9z5
  namespace: e2e-tests-zalando-kube-admission-controller-cvbcz
  ownerReferences:
  - apiVersion: apps/v1
    blockOwnerDeletion: true
```

MULTIPLE CONDITIONS



zalando

MULTIPLE RESOURCE TYPES



LABEL & CUSTOM COLUMNS

labelcols=**application**
customcols=CPU+Requests=**join(', ', spec.containers[*].resources.requests.cpu)**

Pods

Namespace	Name	Application	Ready	Status	Restarts	Age	IP	Node	CPU Usage	Memory Usage	CPU Requests	Memory Requests	Created
default	kube-ops-view-7b9dd46fd8-kmv89	kube-ops-view	1/1	Running	0	13d	10.42.0.54	k3s-demo	11m	37 MiB	50m	50Mi	2019-08-22 07:53:43
default	kube-ops-view-redis-577f846477-lrgk2	kube-ops-view-redis	1/1	Running	0	39d	10.42.0.6	k3s-demo	86m	663 MiB	50m	50Mi	2019-07-27 14:43:17
default	kube-resource-report-5f77c8f5d9-lwtnl	kube-resource-report	2/2	Running	0	24d	10.42.0.44	k3s-demo	0m	36 MiB	5m, 5m	50Mi, 20Mi	2019-08-11 11:37:24
default	kube-web-view-599d9ff485-46w44	kube-web-view	1/1	Running	0	38h	10.42.0.59	k3s-demo	1m	39 MiB	5m	100Mi	2019-09-03 16:29:26
default	nginx-c9767ffdf-22tgk		1/1	Running	0	39d	10.42.0.20	k3s-demo	0m	2 MiB	2m	20M	2019-07-27 20:59:56

uses JMESPath expression

URL STRUCTURE

```
/clusters/{cluster}/namespaces/{namespace}/{resource-type}
```

cluster: cluster aliases separated by "," or "_all"

namespace: namespace name or "_all"

resource-type: plural names separated by "," or "all" (like kubectl get all)

https://kube-web-view.readthedocs.io/

QUERY PARAMETERS FOR LIST VIEW

sort: column name to sort by, optionally with ":desc"

labelcols: additional label columns (comma separated) or "*"

selector: label selector (like kubectl -l), will be passed to Kubernetes API

filter: post-processing filter, either a simple string, Column=Value or Column!=Value

hidecols: hide certain columns

customcols: custom columns using JMESPath

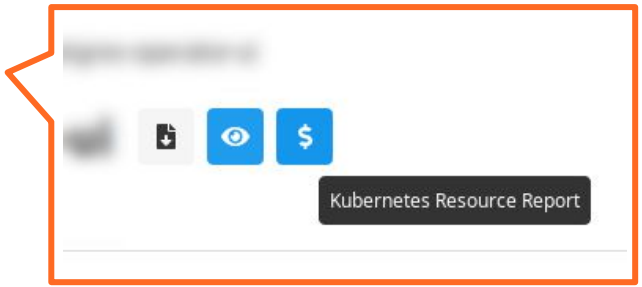
limit: number of rows to show

https://kube-web-view.readthedocs.io/en/latest/features.html#listing-resources

CUSTOMIZATION

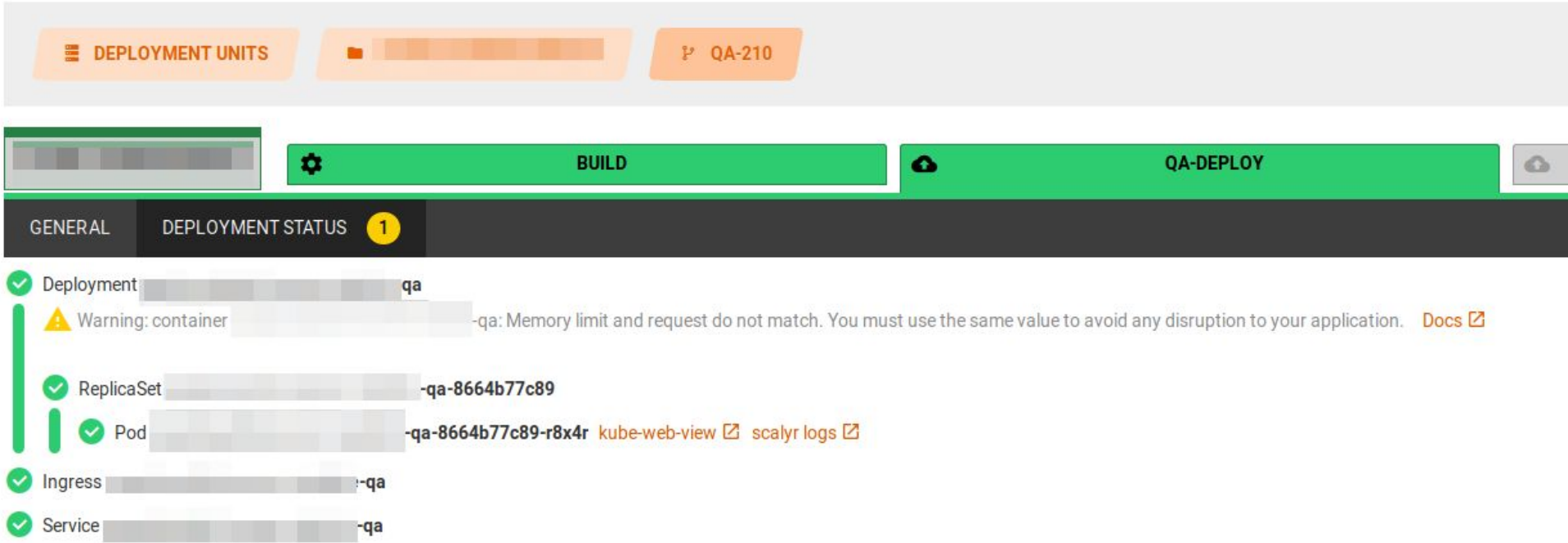
Nearly all aspects can be customized for your org via CLI options:

- External **Links** (monitoring dashboards, ..)
- Default Label and Custom **Columns**
- **Sidebar**, HTML templates
- **OAuth2** integration



kube-web-view.readthedocs.io/en/latest/customization.html

LINKING FROM DEPLOYMENT STATUS



zalando

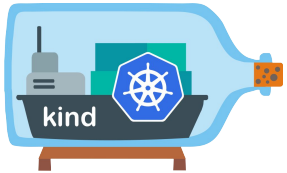
KUBE-WEB-VIEW.DEMO.J-SERV.DE

local / default / pods

Pods

Name	Application	Ready	Status	Restarts	Age
kube-ops-view-7b9dd46fd8-vfqgt	kube-ops-view	1/1	Running	0	46h
kube-ops-view-redis-577f846477-lrgk2	kube-ops-view-redis	1/1	Running	0	15d
kube-resource-report-5f77c8f5d9-lwtnl	kube-resource-report	2/2	Running	0	8h
kube-web-view-75b996b8c8-z2sgt	kube-web-view	1/1	Running	0	57s
nginx-c9767ffdf-22tgk		1/1	Running	0	14d
nginx-c9767ffdf-8qzqr		1/1	Running	0	14d

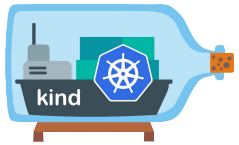
TESTING WITH PYTEST-KIND AND PURE HTML




```python
def test_filter_pods_with_custom_columns(session):
    """
    Test that filtering on custom columns works
    """
    response = session.get(
        "/clusters/local/namespaces/default/pods?customcols=Images=spec.containers[*].image&filter=hjacobs/"
    )
    response.raise_for_status()
    check_links(response, session)
    ths = response.html.find("main table thead th")
    # note: pods have an extra "Links" column (--object-links)
    assert ths[-3].text == "Images"

    rows = response.html.find("main table tbody tr")
    for row in rows:
        cells = row.find("td")
        assert cells[-3].text.startswith("['hjacobs/")
```

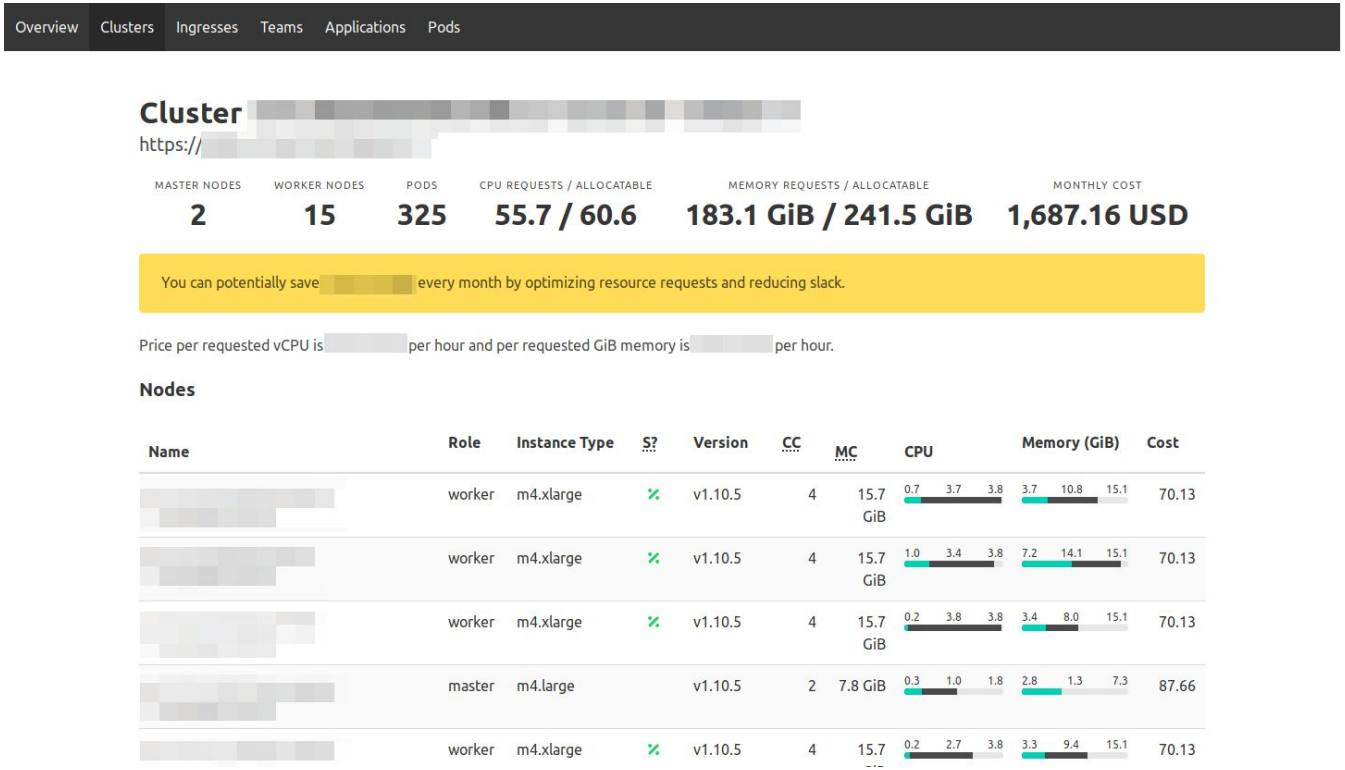
KUBERNETES + PYTHON?

- Pykube
- asyncio / aiohttp
- pytest-kind
- Jinja2 templates



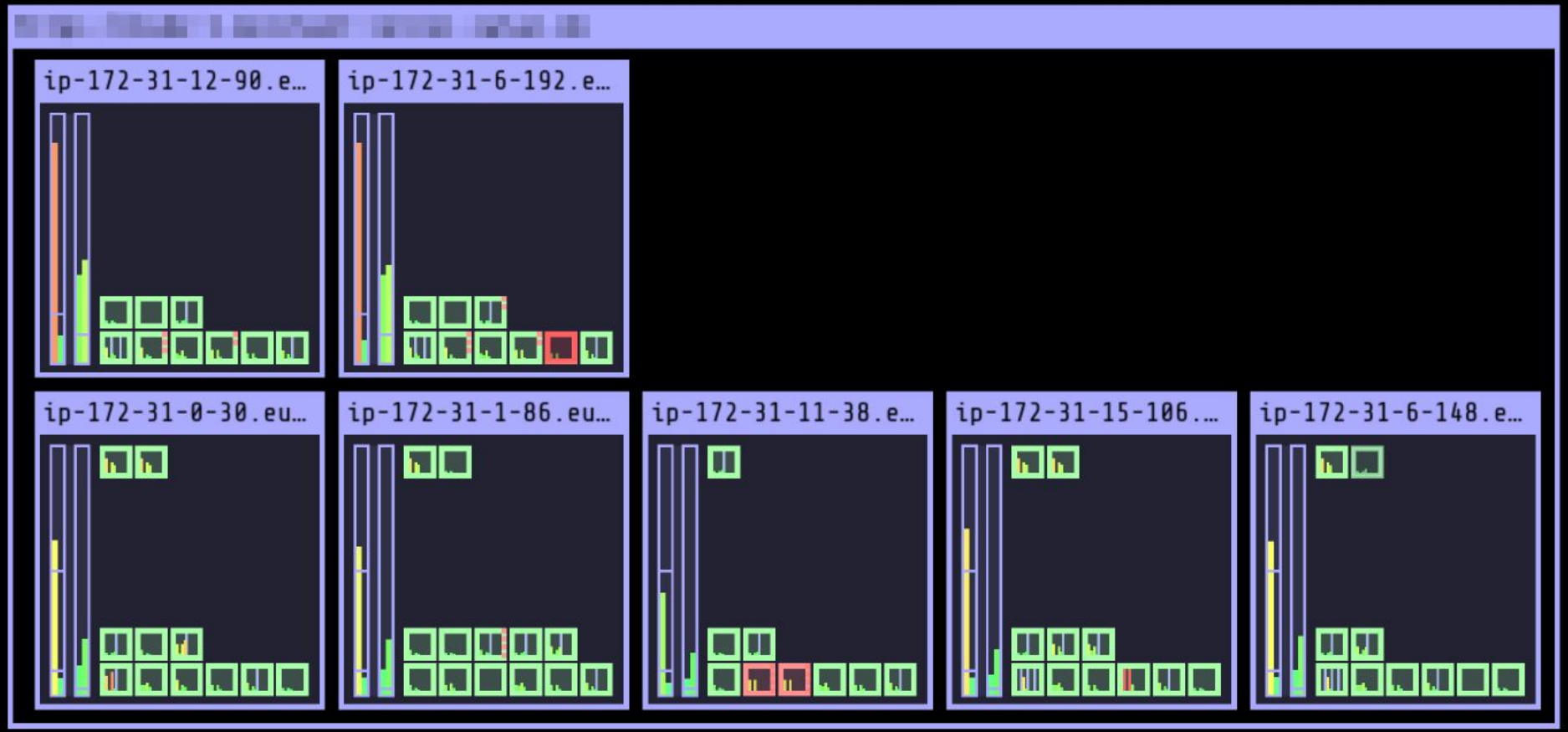
codeberg.org/hjacobs/kube-web-view/

KUBERNETES RESOURCE REPORT



github.com/hjacobs/kube-resource-report

zalando

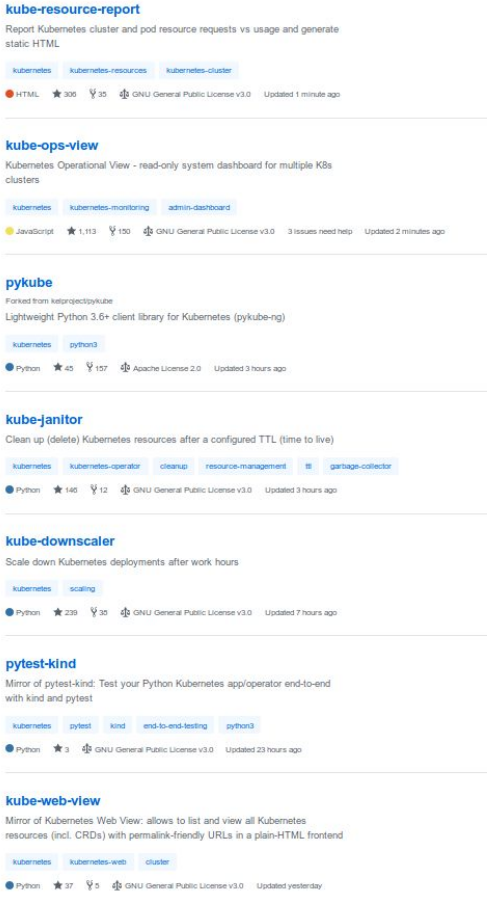


https://github.com/hjacobs/kube-ops-view

https://github.com/hjacobs/kube-ops-view

KUBERNETES + PYTHON = ❤

github.com/hjacobs



zalando

KUBERNETES + PYTHON = ❤

code examples from this talk can be found in

codeberg.org/hjacobs/kubernetes-and-python

ZALANDO OPEN SOURCE

Kubernetes on AWS
github.com/zalando-incubator/kubernetes-on-aws

AWS ALB Ingress controller
github.com/zalando-incubator/kube-ingress-aws-controller

External DNS
github.com/kubernetes-incubator/external-dns

Postgres Operator
github.com/zalando/postgres-operator

Elasticsearch Operator
github.com/zalando-incubator/es-operator

Kubernetes Metrics Adapter
github.com/zalando-incubator/kube-metrics-adapter

zalando

zalando

QUESTIONS?

HENNING JACOBS

henning@zalando.de

@try_except_

Illustrations by @01k