Operator Day 2021

Charmed Operator Development Workshop

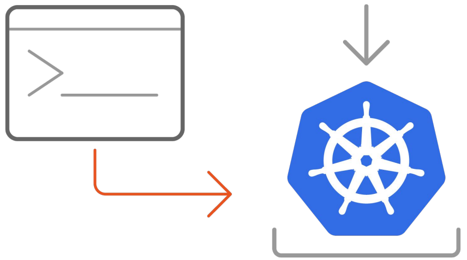
@jnsgruk

CANONICAL ubuntu

Development Setup

```
# Install/Setup MicroK8s
$ sudo snap install --classic microk8s
$ sudo usermod -aG microk8s $(whoami)
$ sudo microk8s status --wait-ready
$ sudo microk8s enable storage dns ingress
$ sudo snap alias microk8s.kubectl kubectl
$ newgrp microk8s

# Install Charmcraft
$ sudo snap install charmcraft

# Install Juju
$ sudo snap install juju --classic

# Bootstrap MicroK8s
$ juju bootstrap microk8s micro
$ juju add-model development
```

Copy and paste from:
jnsgr.uk/demo-gist
jnsgr.uk/demo-slides



Join the charming community

Charmhub

charmhub.io

Forum

discourse.charmhub.io

Chat

chat.charmhub.io

Charmed Operator Development Workshop

Juju Basics

Controllers

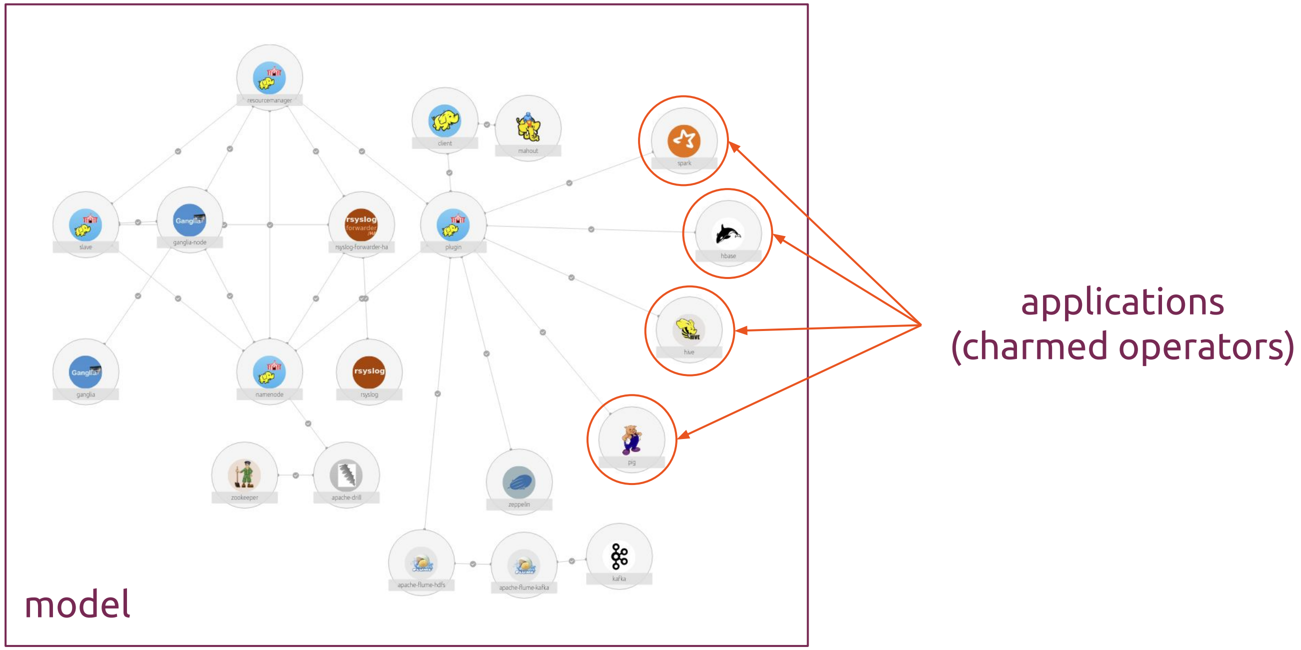
The management node(s) of a deployed Juju cloud environment

Cloud-specific

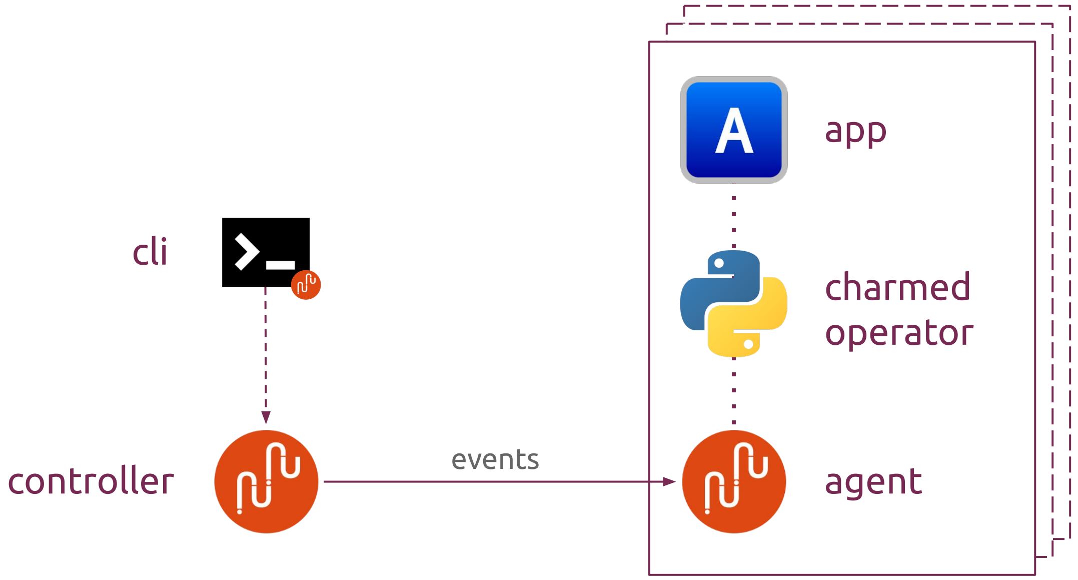
Multi-cloud

Hosted (JAAS)

Models and Applications



Applications and Units



Charmed Operator Development Workshop

Charmed Operator Basics

App domain knowledge, distilled into code

Application code is open source.

Why not share the operations code too?

Charms are ops code, packaged

Lifecycle

install
config
update
remove
scale

Operations / Actions

action_backup
action_restore
action_scan-viruses
action_health-check
action_add-repo
action_reset
action_verify_sigs
action_...
action_...
action_...

Integration

relate_mysql
relate_ldap
relate_proxy
relate_...

Charmed Operators on Kubernetes

```
juju add-model new-model
```

Creates a Kubernetes namespace `'new-model'`

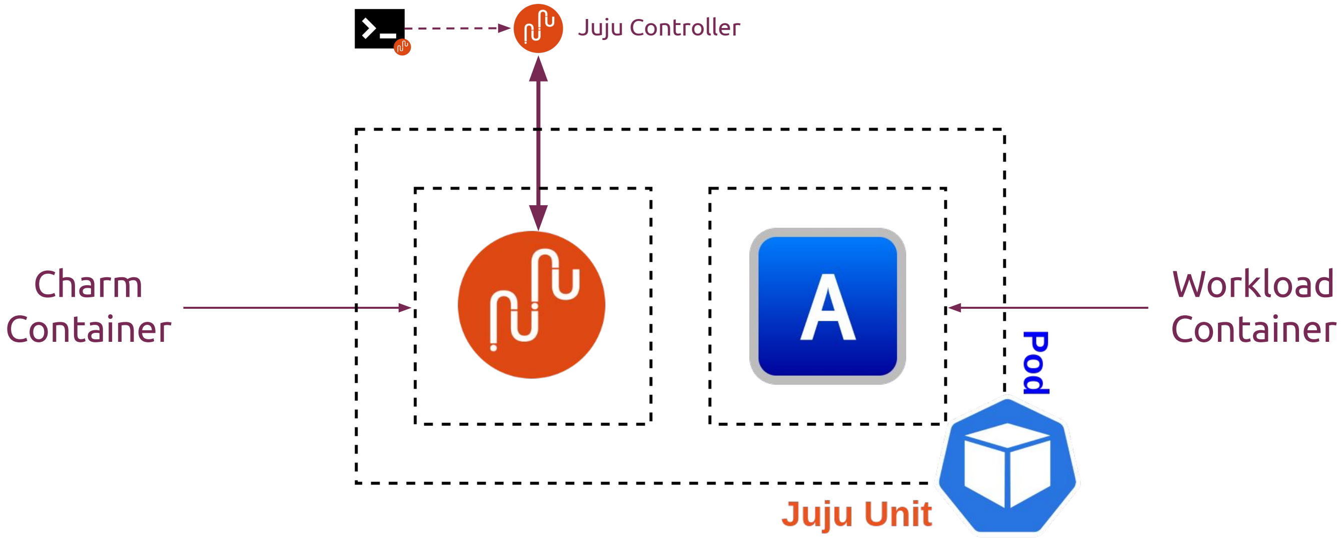
```
juju deploy <application>
```

Create a `StatefulSet` named `'application'` with 1 replica

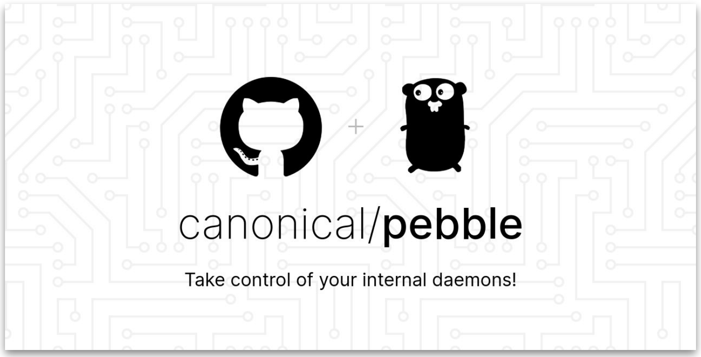
```
juju scale-application <application> 2
```

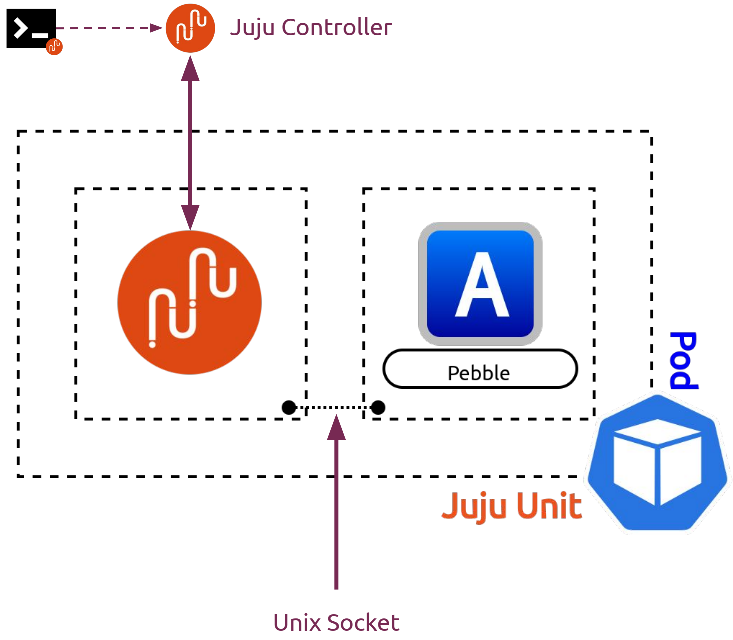
Set # of replicas to 2 for the `StatefulSet`

Charmed Operators on Kubernetes



Pebble


canonical/pebble
Take control of your internal daemons!


Juju Controller
Pebble
Pod
Juju Unit
Unix Socket

Pebble Layers

```yaml
services:
  super-service:
    override: replace
    summary: The super service
    command: /super -a -p 80
    startup: enabled
```

+

```yaml
services:
  super-service:
    override: merge
    environment:
      VAR1: value-1
      VAR2: value-2
```

=

```yaml
services:
  super-service:
    override: replace
    summary: The super service
    command: /super -a -p 80
    startup: enabled
    environment:
      VAR1: value-1
      VAR2: value-2
```

Day 1

Day *n*

Result

Simple operations code, written in Python

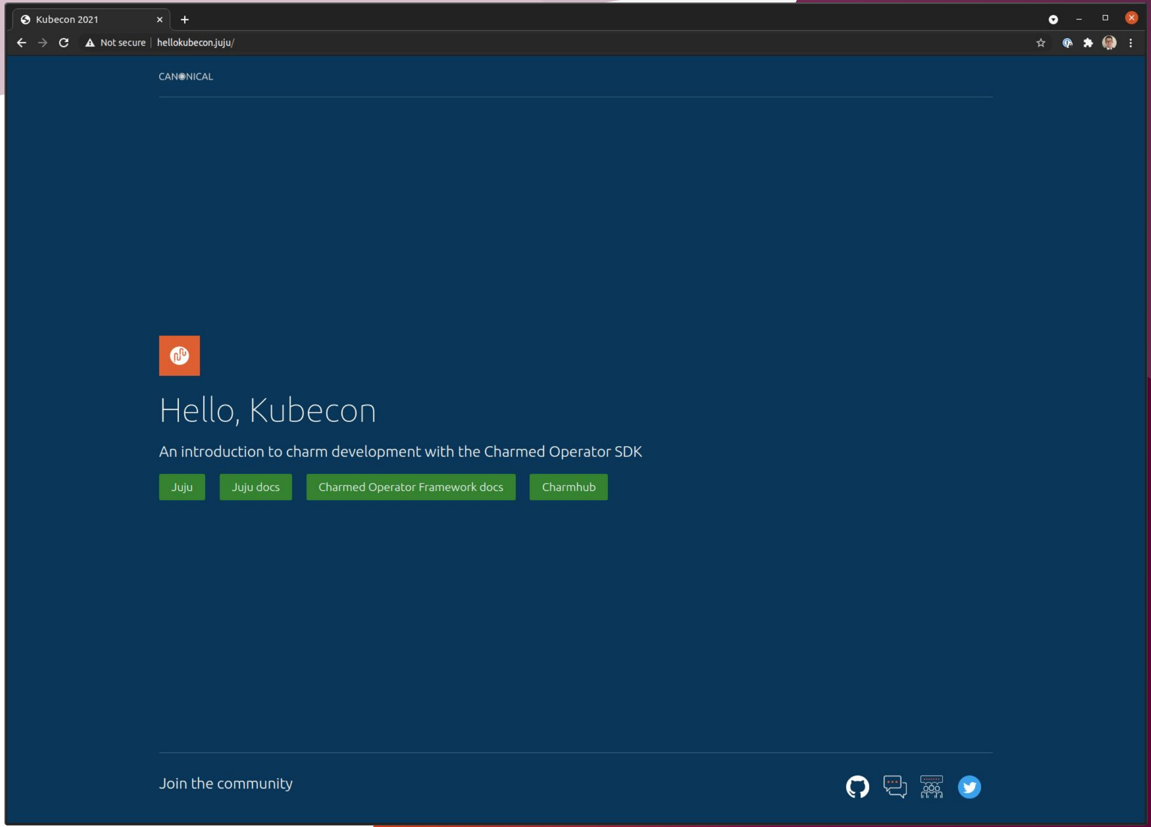
Consistent operator UX and CLI for all operators

Comprehensive unit testing harness

Simplified code-sharing and integration

Workshop outcome

- ✓ Start a workload
- ✓ Handle configuration
- ✓ Day-2 action specification
- ✓ Utilise a charm library
- ✓ Integrate with another application
- ✓ Unit test the operator

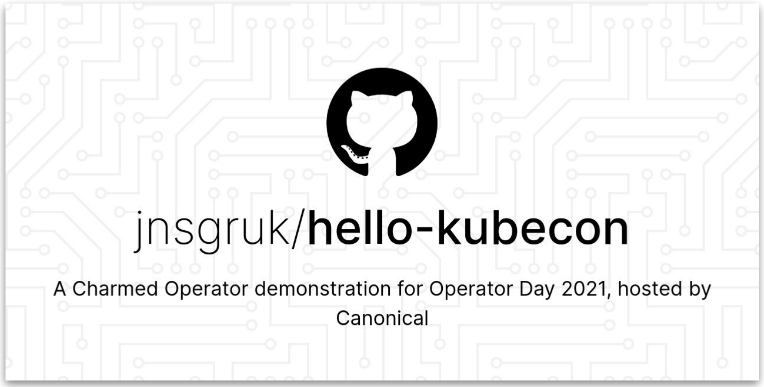


Hello, Kubecon

An introduction to charm development with the Charmed Operator SDK

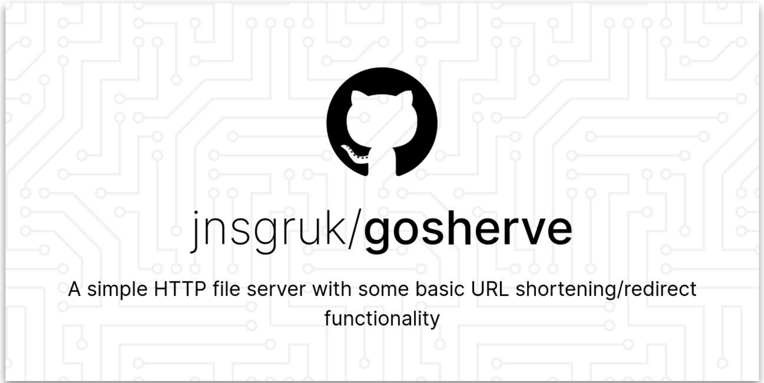
Join the community

Key Information



jnsgruk/**hello-kubecon**

A Charmed Operator demonstration for Operator Day 2021, hosted by Canonical



jnsgruk/**gosherve**

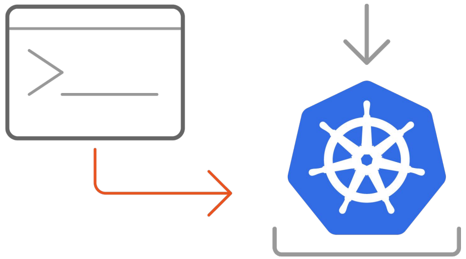
A simple HTTP file server with some basic URL shortening/redirect functionality

jnsgr.uk/demo-gist / jnsgr.uk/demo-slides

Development Setup

```
# Install/Setup MicroK8s
$ sudo snap install --classic microk8s
$ sudo usermod -aG microk8s $(whoami)
$ sudo microk8s status --wait-ready
$ sudo microk8s enable storage dns ingress
$ sudo snap alias microk8s.kubectl kubectl
$ newgrp microk8s

# Install Charmcraft
$ sudo snap install charmcraft

# Install Juju
$ sudo snap install juju --classic

# Bootstrap MicroK8s
$ juju bootstrap microk8s micro
$ juju add-model development
```

Copy and paste from:
jnsgr.uk/demo-gist
jnsgr.uk/demo-slides



Charmed Operator Development Workshop

Charm Initialisation

```
git checkout 1-specify-workload
```

Charmed Operator Initialisation

```
# Create the Charm directory
$ mkdir hello-kubecon; cd hello-kubecon
# Initialise the Charm directory
$ charmcraft init

├── README.md            # The front page documentation for your charm
├── LICENSE              # Your Charm's license, we recommend Apache 2
├── metadata.yaml        # Charmed Operator package description and metadata
├── requirements.txt     # PyPI requirements for the charm runtime environment
├── config.yaml          # Configuration schema for your operator
├── actions.yaml         # Day 2 action declarations, e.g. backup, restore
├── requirements-dev.txt # PyPI requirements for development environment
├── run_tests            # Bash script to run Charm tests
├── src                  # Top-level source code directory for Charm
│   └── charm.py         # Minimal operator using Charmed Operator Framework
└── tests # Top-level directory for Charm tests
    ├── __init__.py
    └── test_charm.py    # Skeleton unit tests for generated charm
```

juju.is/docs/sdk/hello-world

Charmed Operator Initialisation

```
# Ensure that virtualenv support is installed
$ sudo apt update && sudo apt install -y python3-virtualenv
# Create a virtualenv for the charm code
$ virtualenv venv
# Activate the venv
$ source ./venv/bin/activate
# Install dependencies
$ pip install -r requirements-dev.txt
```

juju.is/docs/sdk/hello-world

Starting our workload

metadata.yaml 1-specify-workload

```yaml
# See LICENSE file for licensing details.
name: hello-kubecon
description: |
 A basic demonstration charm that hosts a placeholder webpage with links
 to various Juju/Charmed Operator SDK pages. Hosted using a small, custom
 webserver written in Go (https://github.com/jnsgruk/gosherve). Illustrates
 the use of charm workloads, actions, config, storage and relations.
summary: |
 A demonstration charm for Kubecon Operator Day 2021.

containers:
 gosherve:
   resource: gosherve-image

resources:
 gosherve-image:
   type: oci-image
   description: OCI image for gosherve
```

juju.is/docs/sdk/workloads

Starting our workload

src/charm.py 1-specify-workload

```python
def _on_gosherve_pebble_ready(self, event):
    container = event.workload
    pebble_layer = {
        "summary": "gosherve layer",
        "description": "pebble config layer for gosherve",
        "services": {
            "gosherve": {
                "override": "replace",
                "summary": "gosherve",
                "command": "/gosherve",
                "startup": "enabled",
                "environment": {
                    "REDIRECT_MAP_URL": "https://jnsgr.uk/demo-routes"
                },
            }
        },
    }
    container.add_layer("gosherve", pebble_layer, combine=True)
    container.autostart()
    self.unit.status = ActiveStatus()
```

Test Deployment

Build & Deploy

```
# Build the charm
$ charmcraft pack
# Deploy the charm
$ juju deploy ./hello-kubecon.charm --resource gosherve-image=jnsgruk/gosherve
# Check the Juju status
$ watch -n1 --color juju status --color
```

Check Status

```
Model        Controller  Cloud/Region        Version  SLA          Timestamp
development  micro       microk8s/localhost  2.9.0    unsupported  11:58:51+01:00

App            Version  Status  Scale  Charm          Store  Channel  Rev  OS          Address  Message
hello-kubecon           active      1  hello-kubecon  local             5  kubernetes

Unit              Workload  Agent  Address       Ports  Message
hello-kubecon/0*  active    idle   10.1.215.221
```

Verify

```
$ curl http://10.1.215.221:8080/ops
<a href="https://github.com/canonical/operator">Found</a>.
```

Explore & Troubleshoot Deployment

Juju Debug Log

```
# Set the log level to DEBUG for the development model
$ juju model-config logging-config="<root>=WARNING;unit=DEBUG"
# Follow the debug-log
$ juju debug-log
```

Explore with kubectl

```
$ kubectl -n development get pods
NAME                                READY   STATUS    RESTARTS   AGE
modeloperator-77db8dbbb9-c4rjv      1/1     Running   1          22h
hello-kubecon-0                     2/2     Running   0          9m2s
```

juju.is/docs/help

Charmed Operator Development Workshop

Handling Configuration

```
git checkout 2-handle-configuration
```

Charm Configuration

```
juju config application theme=breeze
```

Handling Configuration

config.yaml   2-handle-configuration

```yaml
# Copyright 2021 Jon Seager
# See LICENSE file for licensing details.
#
# Learn more about config at: https://juju.is/docs/sdk/config

options:
 redirect-map:
   default: https://jnsgr.uk/demo-routes
   description: A URL pointing to a list of redirects for Gosherve.
   type: string
```

juju.is/docs/sdk/config

Handling Configuration

src/charm.py 2-handle-configuration

```
def _gosherve_layer(self):
    """Returns a Pebble configuration layer for Gosherve"""
    return {
        "summary": "gosherve layer",
        "description": "pebble config layer for gosherve",
        "services": {
            "gosherve": {
                "override": "replace",
                "summary": "gosherve",
                "command": "/gosherve",
                "startup": "enabled",
                "environment": {
                    "REDIRECT_MAP_URL": self.config["redirect-map"]
                },
            }
        },
    }
```

github.com/canonical/pebble

Handling Configuration

src/charm.py 2-handle-configuration

```python
def _on_config_changed(self, event):
        """Handle the config-changed event"""
        # Get the gosherve container so we can configure/manipulate it
        container = self.unit.get_container("gosherve")
        # Create a new config layer
        layer = self._gosherve_layer()
        # Get the current config
        services = container.get_plan().to_dict().get("services", {})
        # Check if there are any changes to services
        if services != layer["services"]:
            # Changes were made, add the new layer
            container.add_layer("gosherve", layer, combine=True)
            logging.info("Added updated layer 'gosherve' to Pebble plan")
            # Stop the service if it is already running
            if container.get_service("gosherve").is_running():
                container.stop("gosherve")
            # Restart it and report a new status to Juju
            container.start("gosherve")
            logging.info("Restarted gosherve service")
        # All is well, set an ActiveStatus
        self.unit.status = ActiveStatus()
```

juju.is/docs/sdk/events

Test Deployment

Build & Deploy

```
# Build the charm
charmcraft pack
# Deploy the charm
juju refresh hello-kubecon --path=./hello-kubecon.charm
# Check the juju status - note the blocked status
watch -n1 --color juju status --color
```

Set Configuration

```
# Change the configuration
$ juju config hello-kubecon
redirect-map="https://jnsgr.uk/demo-routes-alt"
# Check the juju status - note the active status
$ watch -n1 --color juju status --color
```

juju.is/docs/sdk/hello-world

Charmed Operator Development Workshop

Handling Storage

```
git checkout 3-storage
```

Handling Storage

metadata.yaml 3-storage

```yaml
# ...

containers:
 gosherve:
   resource: gosherve-image
   mounts:
     - storage: webroot
       location: /srv

# ...

storage:
 webroot:
   type: filesystem
   location: /srv
```

juju.is/docs/sdk/storage

Handling Storage

src/charm.py 3-storage

```python
def _gosherve_layer(self):
    """Returns a Pebble configuration layer for Gosherve"""
    return {
        "summary": "gosherve layer",
        "description": "pebble config layer for gosherve",
        "services": {
            "gosherve": {
                "override": "replace",
                "summary": "gosherve",
                "command": "/gosherve",
                "startup": "enabled",
                "environment": {
                    "REDIRECT_MAP_URL": self.config["redirect-map"],
                    "WEBROOT": "/srv",
                },
            }
        },
    }
```

juju.is/docs/sdk/storage

Handling Storage

src/charm.py   3-storage

```python
class HelloKubeconCharm(CharmBase):
    """Charm the service."""

    def __init__(self, *args):
        super().__init__(*args)
        self.framework.observe(self.on.install, self._on_install)
        self.framework.observe(self.on.config_changed, self._on_config_changed)

    def _on_install(self, _):
        # Download the site
        self._fetch_site()
```

juju.is/docs/sdk/storage

Fetching the demo site

src/charm.py 3-storage

```
def _fetch_site(self):
        """Fetch latest copy of website from Github and move into webroot"""
        # Set the site URL
        site_src = "https://jnsgr.uk/demo-site"
        # Set some status and do some logging
        self.unit.status = MaintenanceStatus("Fetching web site")
        logger.info("Downloading site from %s", site_src)
        # Download the site
        urllib.request.urlretrieve(site_src, "/srv/index.html")
        # Set the unit status back to Active
        self.unit.status = ActiveStatus()
```

juju.is/docs/sdk/storage

Test Deployment

Build & Deploy

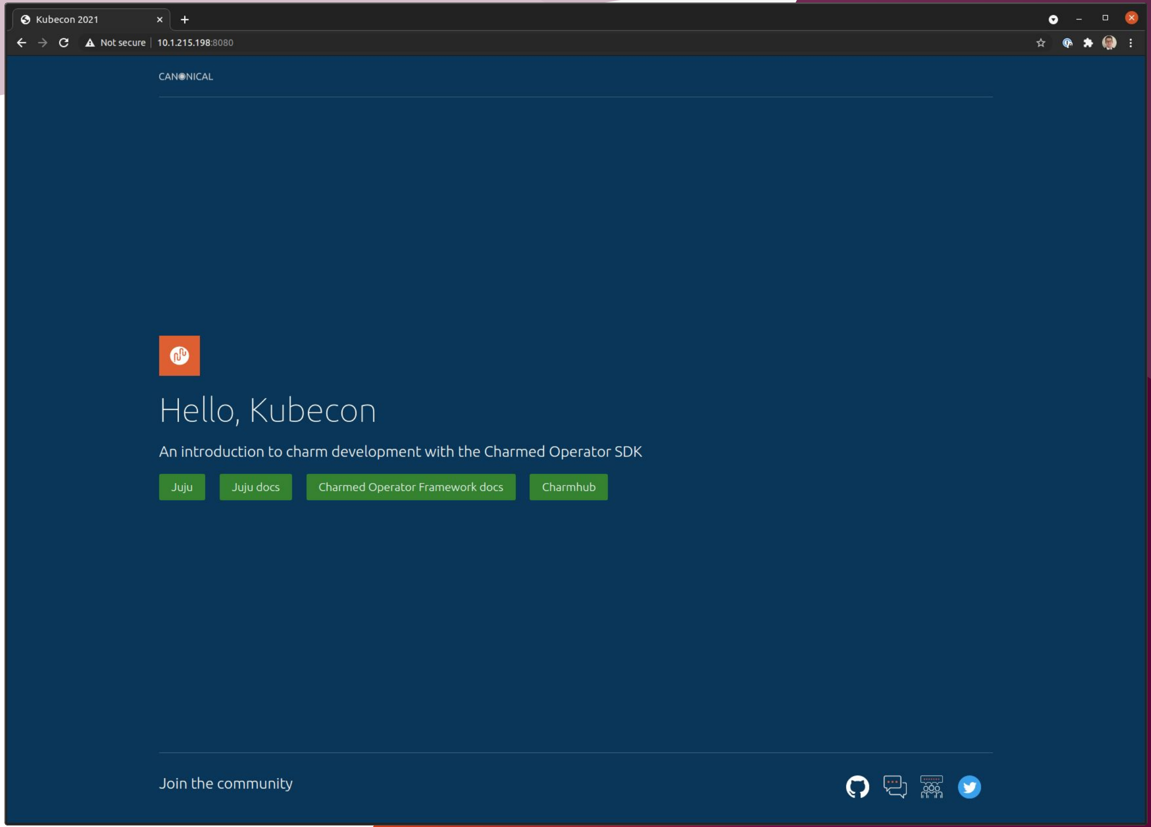
```
# Build the charm
$ charmcraft pack
# Remove old charm (cannot refresh -- storage)
$ juju remove-application hello-kubecon
# Redeploy
$ juju deploy ./hello-kubecon.charm --resource gosherve-image=jnsgruk/gosherve
# Check the juju status
$ watch -n1 --color juju status --color
```

Verify

```
# Verify
$ curl "http://<ip-address>:8080/"
```

Explore

```
# Explore
$ kubectl -n development describe pod hello-kubecon-0
```



Kubecon 2021

Not secure | 10.1.215.198:8080

CANONICAL

Hello, Kubecon

An introduction to charm development with the Charmed Operator SDK

Join the community

Charmed Operator Development Workshop

Implementing an action

```
git checkout 4-action
```

Implementing an action

actions.yaml 4-action

```
# Copyright 2021 Jon Seager
# See LICENSE file for licensing details.
#
# Learn more about actions at: https://juju.is/docs/sdk/actions

pull-site:
 description: Unpack latest version of website from remote source.
```

juju.is/docs/sdk/actions

Implementing an action

src/charm.py 4-action

```python
# ...
def __init__(self, *args):
    super().__init__(*args)
    self.framework.observe(self.on.install, self._on_install)
    self.framework.observe(self.on.config_changed, self._on_config_changed)
    self.framework.observe(self.on.pull_site_action, self._pull_site_action)

# ...

def _pull_site_action(self, event):
    """Action handler that pulls the latest site archive and unpacks it"""
    self._fetch_site()
    event.set_results({"result": "site pulled"})

# ...
```

juju.is/docs/sdk/actions

Test Deployment

Build & Deploy

```
# Build the charm
$ charmcraft pack
# Refresh the deployment
$ juju refresh hello-kubecon --path=./hello-kubecon.charm
# Check the juju status
$ watch -n1 --color juju status --color
```

Run the 'pull-site' action

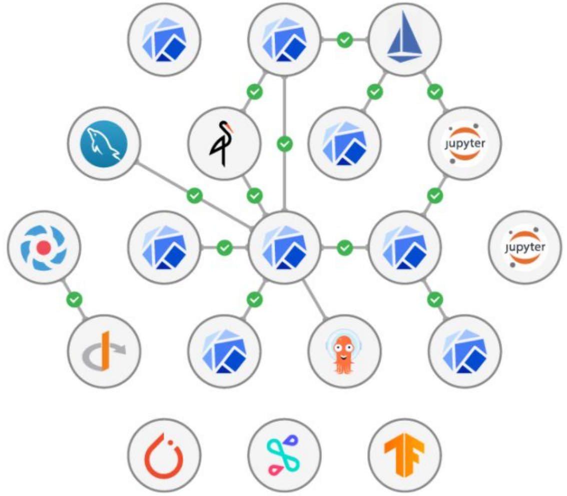
```
# Use the new Juju Actions UX
$ export JUJU_FEATURES=actions-v2
# Run the action
$ juju run hello-kubecon/0 pull-site --format=yaml
Running operation 1 with 1 task
  - task 2 on hello-kubecon/0

Waiting for task 2...
hello-kubecon/0:
  id: "2"
  results:
# ...
```

Charmed Operator Development Workshop

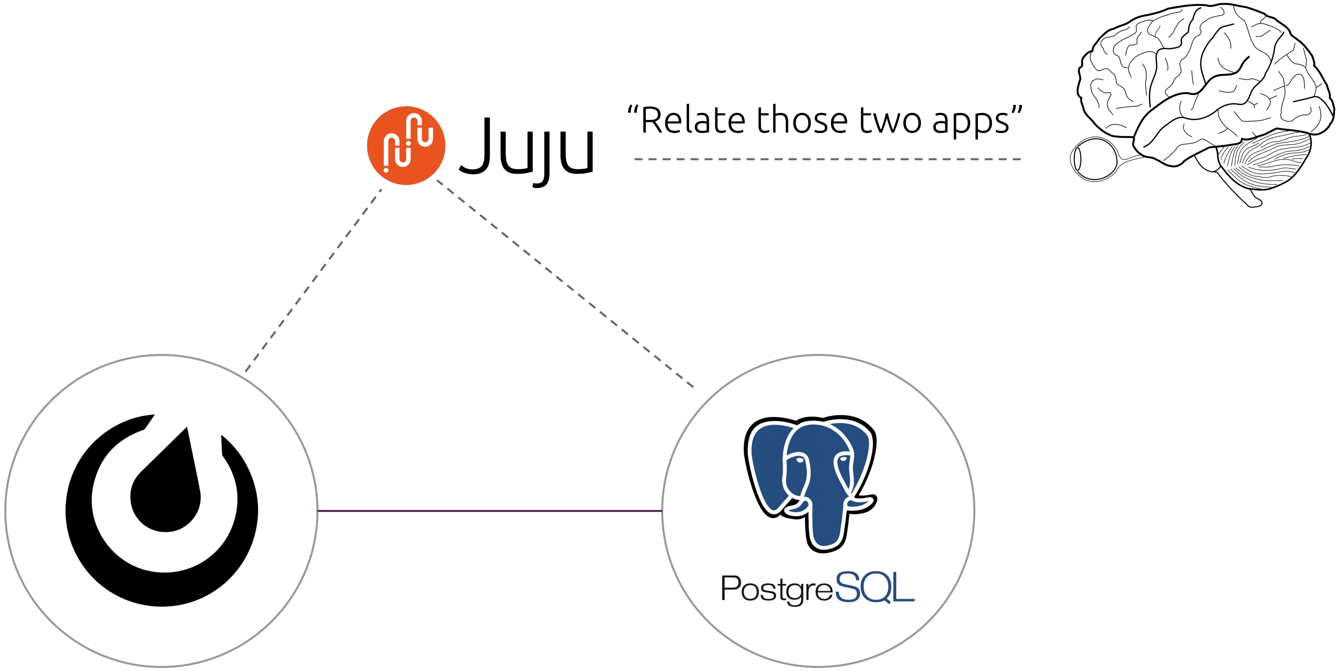
Relations, Libraries, Ingress

```
git checkout 5-ingress
```

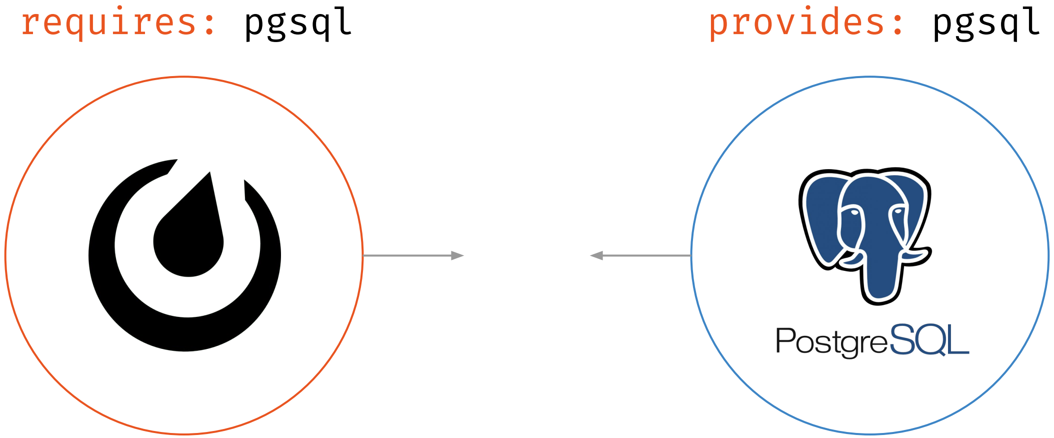
Relations



Relations

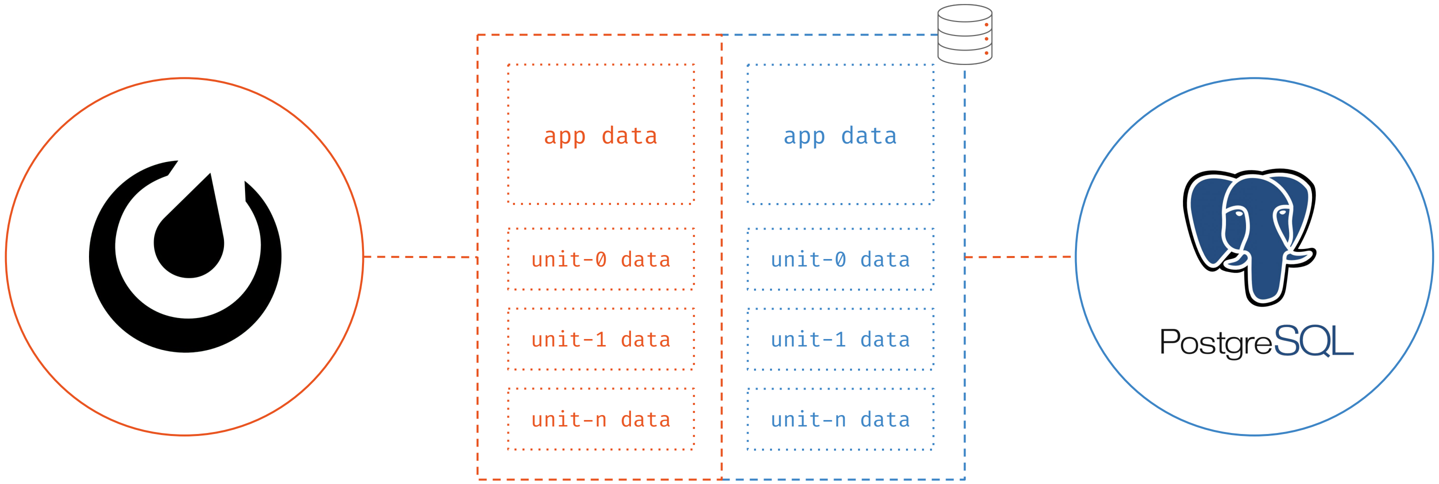


Relations

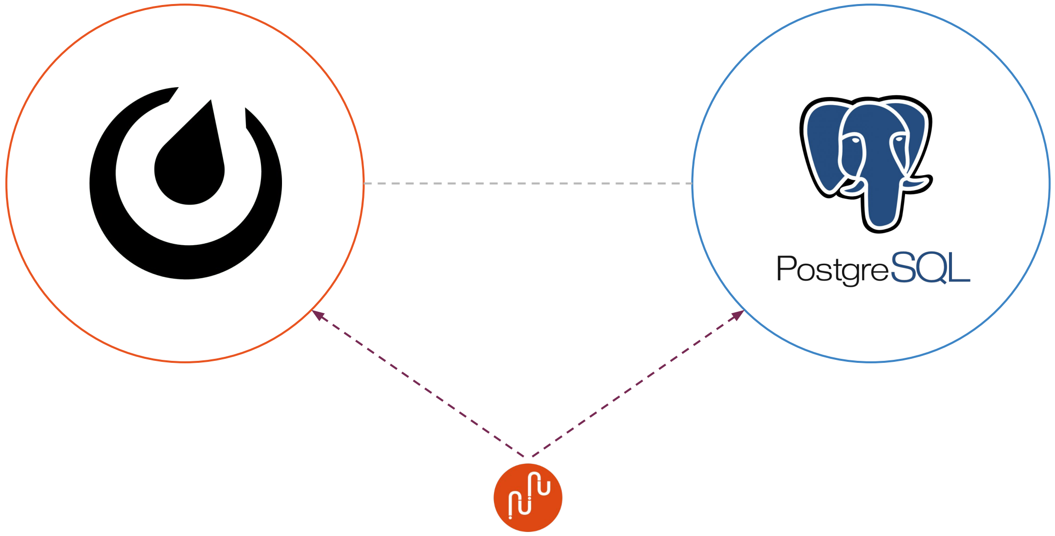


```
juju relate mattermost postgresql
```

Relations

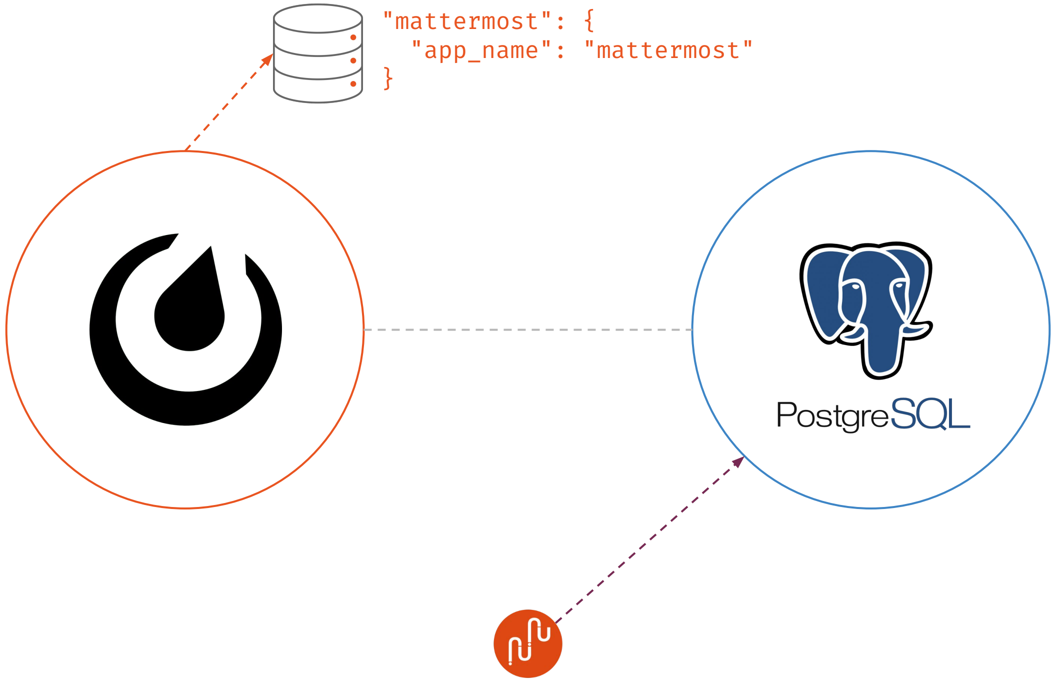

app data
unit-0 data
unit-1 data
unit-n data
app data
unit-0 data
unit-1 data
unit-n data
PostgreSQL

Relations



relation-joined

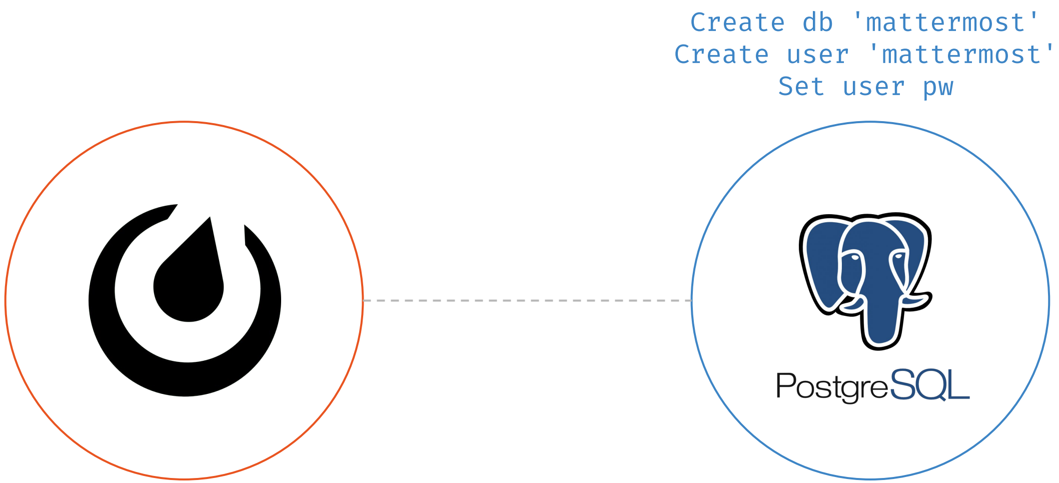
Relations



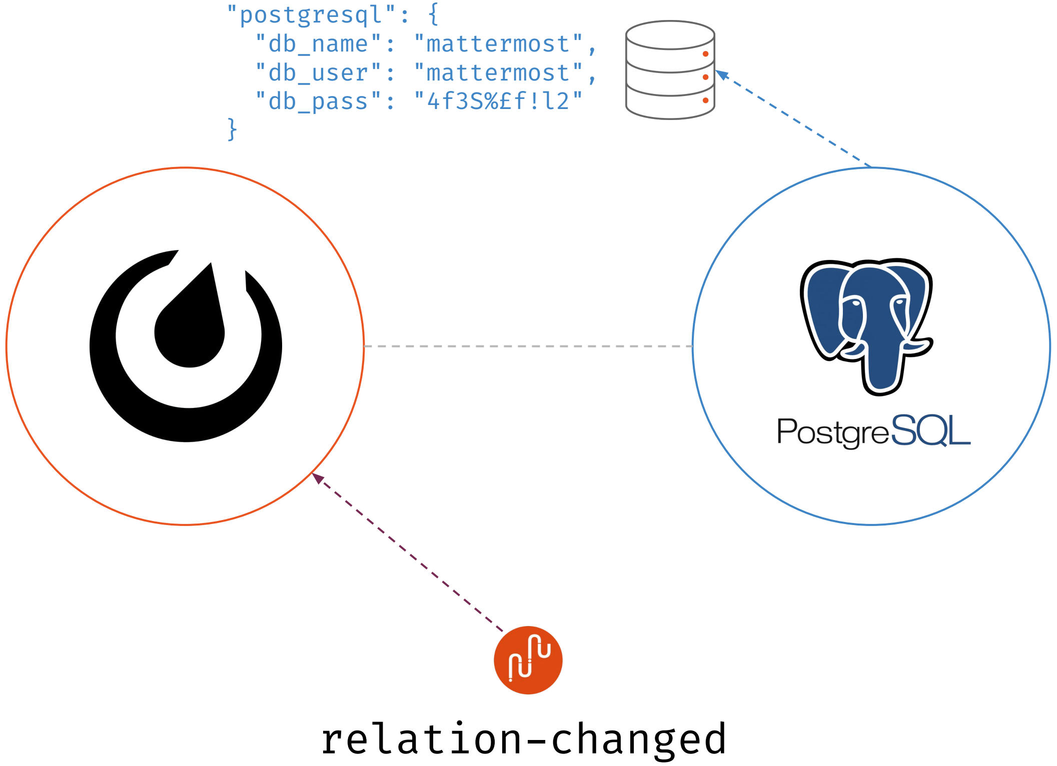
```
"mattermost": {
  "app_name": "mattermost"
}
```

relation-changed

Relations

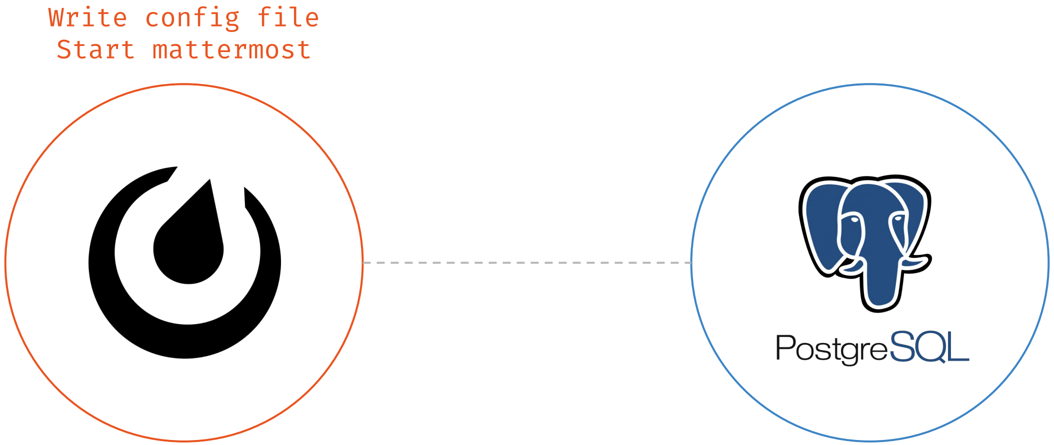

Create db 'mattermost'
Create user 'mattermost'
Set user pw
PostgreSQL

Relations

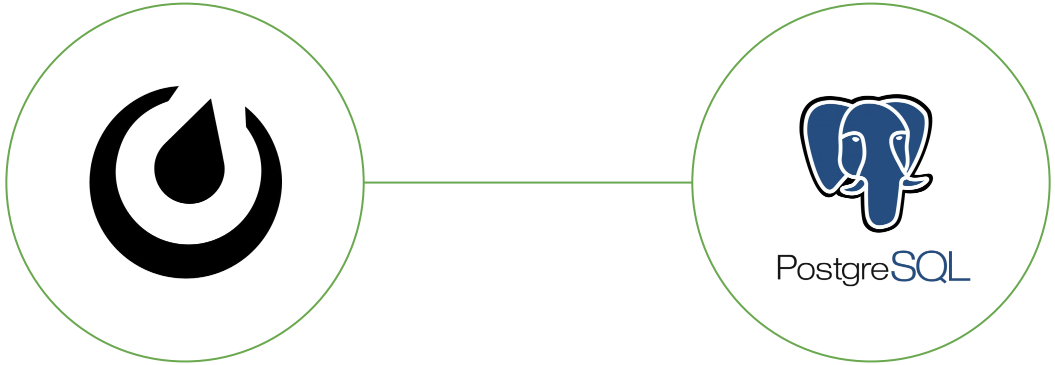


```
"postgresql": {
  "db_name": "mattermost",
  "db_user": "mattermost",
  "db_pass": "4f3S%£f!l2"
}
```

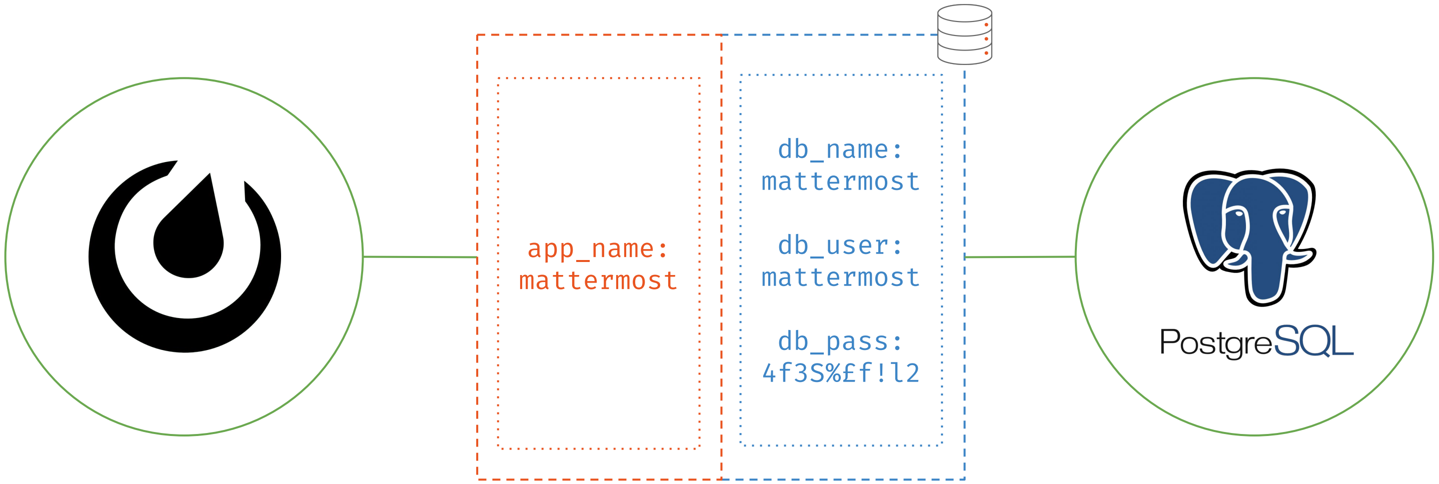
Relations



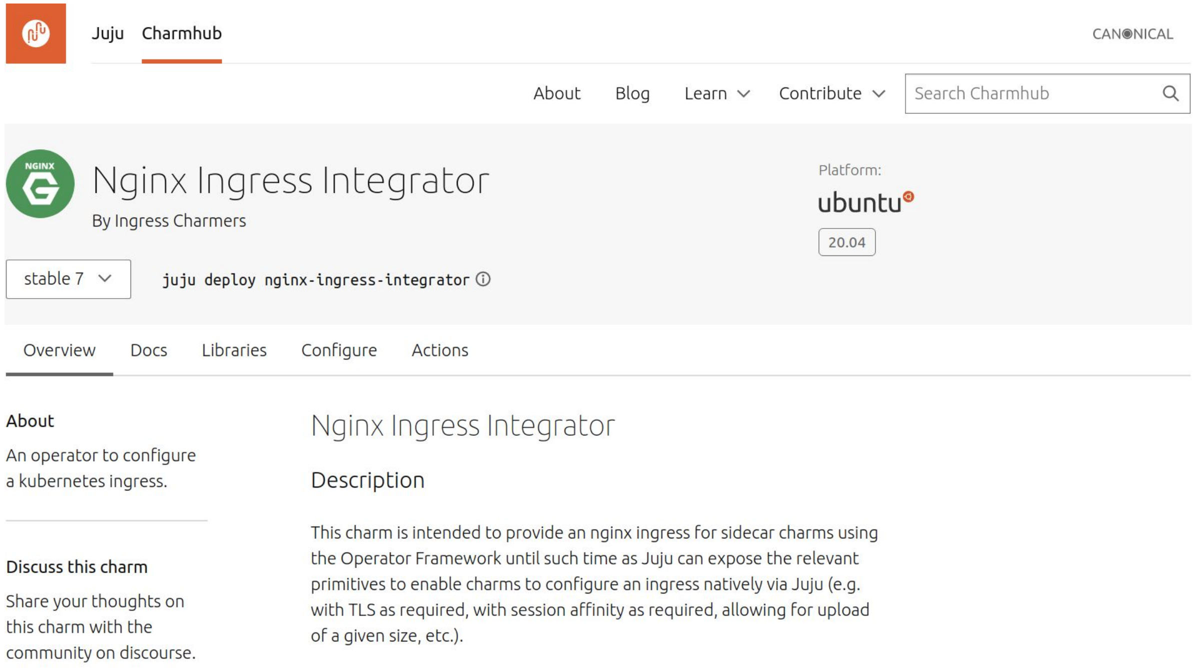
Relations



Relations


app_name:
mattermost
db_name:
mattermost
db_user:
mattermost
db_pass:
4f3S%£f!l2
PostgreSQL

Ingress Charm



Nginx Ingress Integrator

By Ingress Charmers

Platform: ubuntu 20.04

stable 7

```
juju deploy nginx-ingress-integrator
```

About

An operator to configure a kubernetes ingress.

Discuss this charm

Share your thoughts on this charm with the community on discourse.

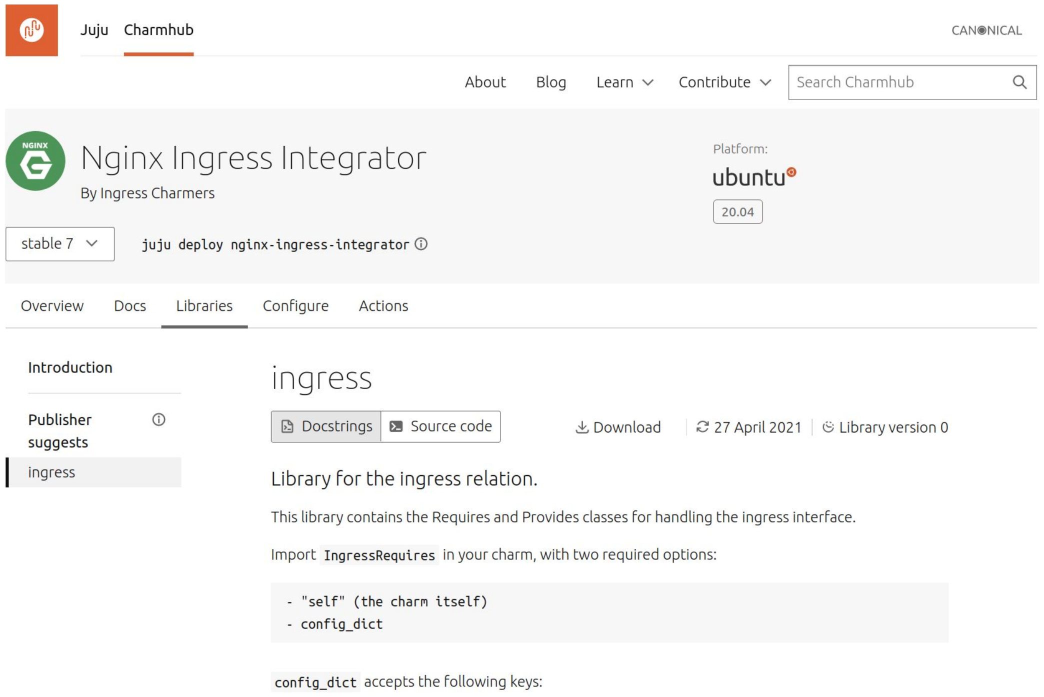
Nginx Ingress Integrator

Description

This charm is intended to provide an nginx ingress for sidecar charms using the Operator Framework until such time as Juju can expose the relevant primitives to enable charms to configure an ingress natively via Juju (e.g. with TLS as required, with session affinity as required, allowing for upload of a given size, etc.).

charmhub.io/nginx-ingress-integrator

Libraries



Nginx Ingress Integrator

By Ingress Charmers

Platform: ubuntu 20.04

stable 7

```
juju deploy nginx-ingress-integrator
```

ingress

27 April 2021 | Library version 0

Library for the ingress relation.

This library contains the Requires and Provides classes for handling the ingress interface.

Import `IngressRequires` in your charm, with two required options:

```
- "self" (the charm itself)
- config_dict
```

`config_dict` accepts the following keys:

charmhub.io/nginx-ingress-integrator

Relations, Libraries and Ingress

Fetch and integrate the library

```
# Check the there is a library associated with the charm
$ charmcraft list-lib nginx-ingress-integrator
# Fetch the library into our charm
$ charmcraft fetch-lib charms.nginx_ingress_integrator.v0.ingress
# Ensure library was imported into the correct place
$ ls -l lib/charms/nginx_ingress_integrator/v0/
```

juju.is/docs/sdk/libraries

Relations, Libraries and Ingress

src/metadata.yaml 5-ingress

```yaml
name: hello-kubecon
description: |
 A basic demonstration charm that hosts a placeholder webpage with links
 to various Juju/Charmed Operator SDK pages. Hosted using a small, custom
 webserver written in Go (https://github.com/jnsgruk/gosherve). Illustrates
 the use of charm workloads, actions, config, storage and relations.
summary: |
 A demonstration charm for Kubecon Operator Day 2021.

# ...

requires:
 ingress:
   interface: ingress
```

juju.is/docs/sdk/relations

Relations, Libraries and Ingress

src/charm.py 5-ingress

```python
from charms.nginx_ingress_integrator.v0.ingress import IngressRequires
# ...

class HelloKubeconCharm(CharmBase):
    """Charm the service."""

    def __init__(self, *args):
        super().__init__(*args)
        self.framework.observe(self.on.install, self._on_install)
        self.framework.observe(self.on.config_changed, self._on_config_changed)
        self.framework.observe(self.on.pull_site_action, self._pull_site_action)

        self.ingress = IngressRequires(self, {
            "service-hostname": "hellokubecon.juju",
            "service-name": self.app.name,
            "service-port": 8080
        })
```

juju.is/docs/sdk/libraries

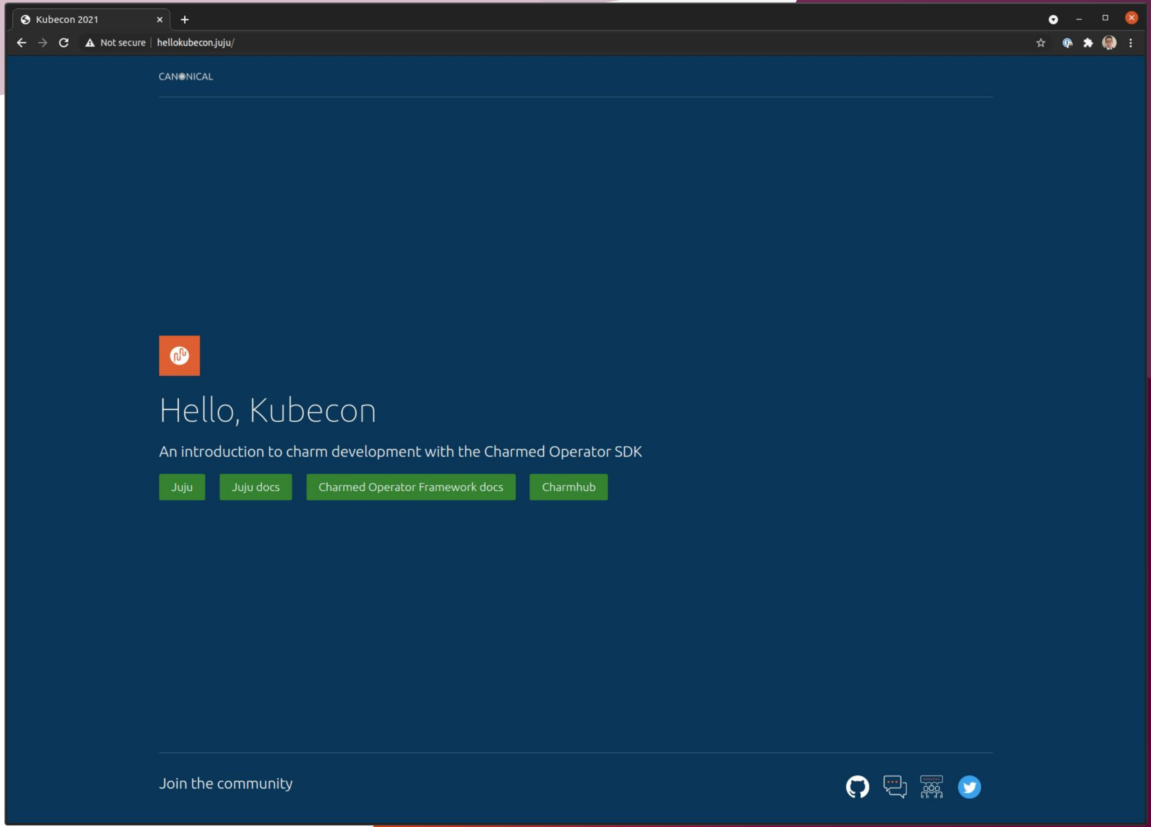
Test Deployment

Build & Deploy

```
# Build the charm
$ charmcraft pack
# Refresh the deployment
$ juju refresh hello-kubecon --path=./hello-kubecon.charm
```

Deploy and Integrate Ingress

```
# Deploy the nginx-ingress-integrator
$ juju deploy nginx-ingress-integrator
# Relate our application to the ingress integrator
$ juju relate hello-kubecon nginx-ingress-integrator
# Set the ingress class for microk8s
$ juju config nginx-ingress-integrator ingress-class="public"
# Add an entry to our hosts file
$ echo "127.0.1.1 hellokubecon.juju" | sudo tee -a /etc/hosts
# Check the juju status
$ watch -n1 --color juju status --color
```


Kubecon 2021

Not secure | hellokubecon.juju/

CANONICAL

Hello, Kubecon

An introduction to charm development with the Charmed Operator SDK

Join the community

Charmed Operator Development Workshop

Unit Testing

```
git checkout master
```

Unit Testing

tests/test_charm.py

master

```
def test_gosherve_layer(self):
    # Test with empty config.
    self.assertEqual(
        self.harness.charm.config['redirect-map'], "https://jnsgr.uk/demo-routes"
    )
    expected = {
        "summary": "gosherve layer",
        "description": "pebble config layer for gosherve",
        "services": {
            "gosherve": {
                # ...
                },
            }
        },
    }
    self.assertEqual(self.harness.charm._gosherve_layer(), expected)
    # ...
```

juju.is/docs/sdk/testing

Unit Testing

tests/test_charm.py master

```python
def test_on_config_changed(self):
    plan = self.harness.get_container_pebble_plan("gosherve")
    self.assertEqual(plan.to_dict(), {})
    # Trigger a config-changed hook. Since there was no plan initially, the
    # "gosherve" service in the container won't be running so we'll be
    # testing the `is_running() == False` codepath.
    self.harness.update_config({"redirect-map": "test value"})
    plan = self.harness.get_container_pebble_plan("gosherve")
    # Get the expected layer from the gosherve_layer method (tested above)
    expected = self.harness.charm._gosherve_layer()
    expected.pop("summary", "")
    expected.pop("description", "")
    # Check the plan is as expected
    self.assertEqual(plan.to_dict(), expected)
    self.assertEqual(self.harness.model.unit.status, ActiveStatus())
    container = self.harness.model.unit.get_container("gosherve")
    self.assertEqual(container.get_service("gosherve").is_running(), True)
```

juju.is/docs/sdk/testing

Test Deployment

Run the tests

```
$ ./run_tests
test_gosherve_layer (tests.test_charm.TestCharm) ... ok
test_on_config_changed (tests.test_charm.TestCharm) ... ok

----------------------------------------------------------------------
Ran 2 tests in 0.003s

OK
Name            Stmts   Miss  Cover   Missing
--------------------------------------------
src/charm.py       51      8    84%   46, 104-109, 113-114, 118
--------------------------------------------
TOTAL              51      8    84%
```

Workshop outcome - recap!

- ✓ Start a workload
- ✓ Handle configuration
- ✓ Day-2 action specification
- ✓ Utilise a charm library
- ✓ Integrate with another application
- ✓ Unit test the operator

Simple operations code, written in Python

Consistent operator UX and CLI for all operators

Comprehensive unit testing harness

Simplified code-sharing and integration

Join the charming community

Charmhub

charmhub.io

Forum

discourse.charmhub.io

Chat

chat.charmhub.io

Thank you