

From ChartMuseum to Harbor

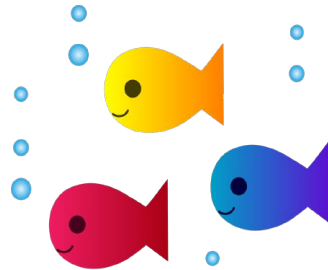
Helm Summit
Amsterdam, September 2019

Josh Dolitsky
>> bloodorange.io

@jdolitsky

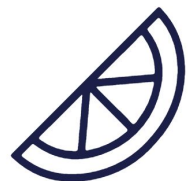


I'm Josh Dolitsky



- ▶ Founder / Engineer - **bloodorange.io**
- ▶ Creator / Core Maintainer - **ChartMuseum**
- ▶ Org / Core Maintainer - **Helm**
- ▶ Here to tell you a story and predict the future

Who likes lunch?



bloodorange.io

Proud sponsor of your lunch today!

*(We hope that it tastes good and
meets all dietary requirements!)*



@jdolitsky

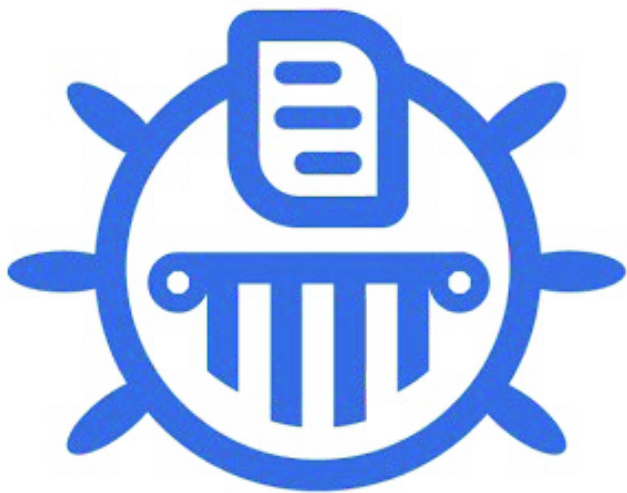
Today I shall make a prediction

Not too bold of a prediction, I don't think..



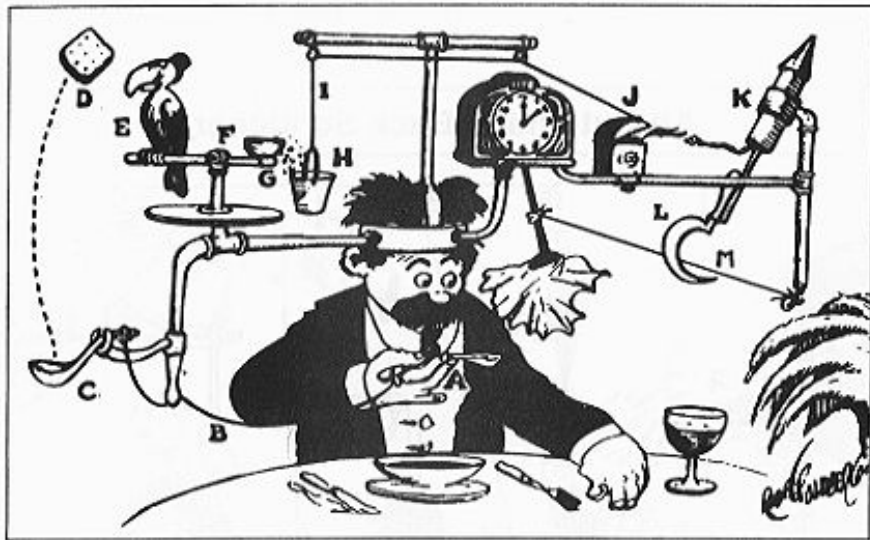
By 2021, **OCI Distribution**
will become the standard
for sharing/storing things
across all cloud-native tools
(things such as Helm charts)

The title of this talk is...



But really it should be...

Self-Operating Napkin



OPEN CONTAINER
INITIATIVE

Let's get back to that in a moment.



ChartMuseum, you said?





The ChartMuseum Project

- ▶ Web server written in Go, started in 2017 (for fun)
- ▶ Donated to Helm as an official subproject
- ▶ Simple, easy way to run a chart repository
- ▶ Pluggable storage options: S3, Azure, Google, etc.
- ▶ Powers lots of companies' internal chart services



CHARTMUSEUM

github.com/helm/chartmuseum



@jdolitsky

**“The chart repository
system is insufficient”**



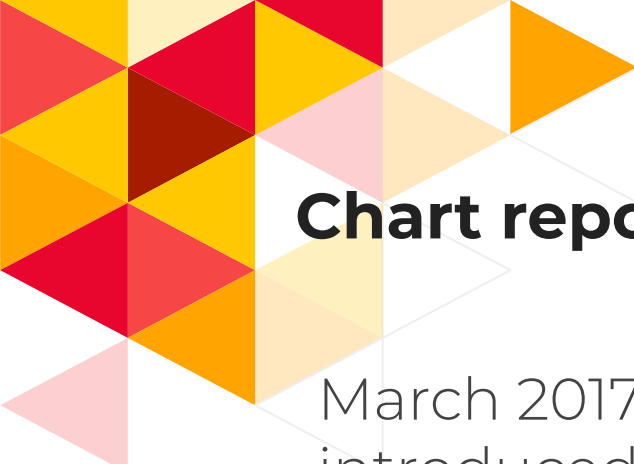
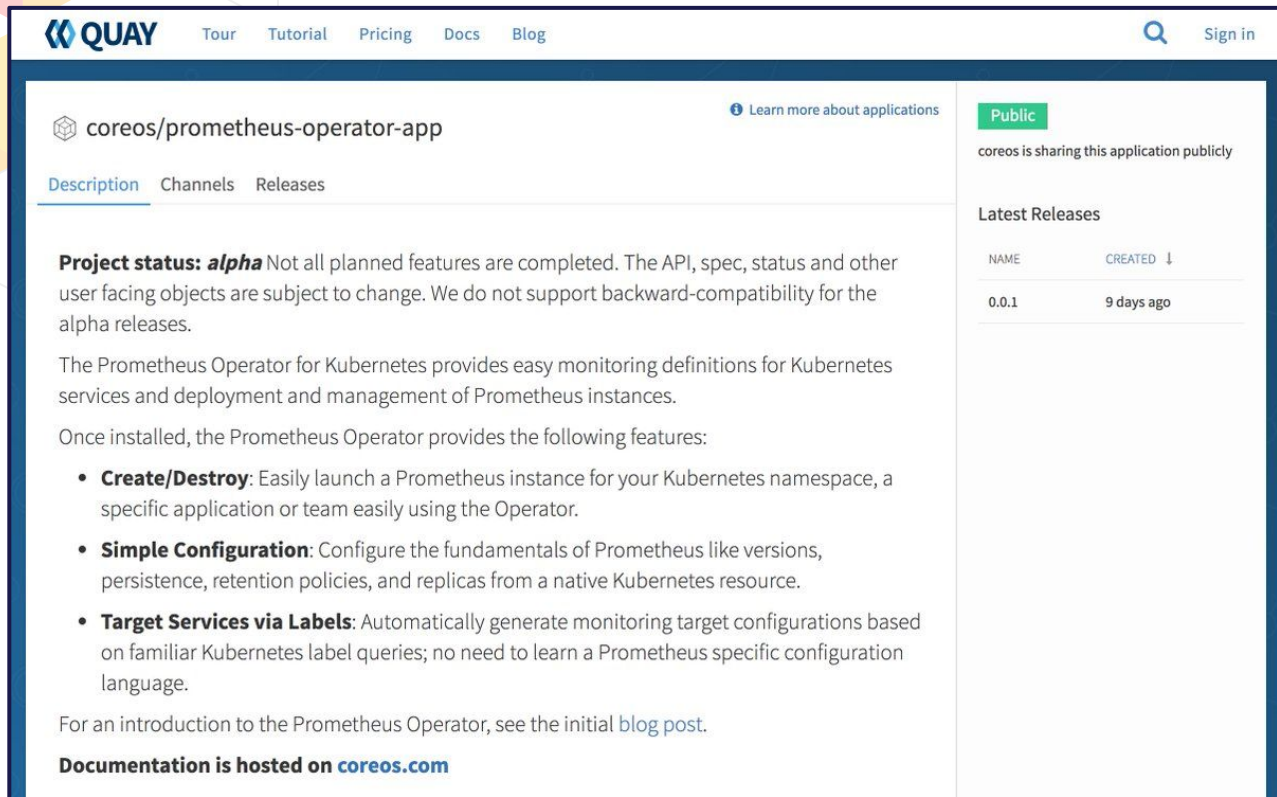


Chart repos fall short (pt. 1)

March 2017, Quay App Registry & Helm plugin introduced as an alternative to using chart repos

- ▶ Led by Antoine Legrand & Jimmy Zelinskie
- ▶ API layer on top of a container registry (Quay.io)
- ▶ Features such as customizable release channels

Quay App Registry



The screenshot shows the Quay App Registry interface. At the top, there's a navigation bar with the Quay logo, links for Tour, Tutorial, Pricing, Docs, and Blog, a search icon, and a Sign in link. The main content area displays the application 'coreos/prometheus-operator-app'. It has tabs for Description (selected), Channels, and Releases. The description section includes a 'Project status: alpha' warning, a paragraph about the Prometheus Operator for Kubernetes, and a list of features: Create/Destroy, Simple Configuration, and Target Services via Labels. A 'Public' badge is visible, indicating the application is publicly shared. On the right, there's a 'Latest Releases' section with a table showing the version 0.0.1 released 9 days ago.

coreos/prometheus-operator-app [Learn more about applications](#)

Public
coreos is sharing this application publicly

[Description](#) [Channels](#) [Releases](#)

Project status: *alpha* Not all planned features are completed. The API, spec, status and other user facing objects are subject to change. We do not support backward-compatibility for the alpha releases.

The Prometheus Operator for Kubernetes provides easy monitoring definitions for Kubernetes services and deployment and management of Prometheus instances.

Once installed, the Prometheus Operator provides the following features:

- **Create/Destroy:** Easily launch a Prometheus instance for your Kubernetes namespace, a specific application or team easily using the Operator.
- **Simple Configuration:** Configure the fundamentals of Prometheus like versions, persistence, retention policies, and replicas from a native Kubernetes resource.
- **Target Services via Labels:** Automatically generate monitoring target configurations based on familiar Kubernetes label queries; no need to learn a Prometheus specific configuration language.

For an introduction to the Prometheus Operator, see the initial [blog post](#).

Documentation is hosted on [coreos.com](#)

Latest Releases

NAME	CREATED ↓
0.0.1	9 days ago

Chart repos fall short (pt. 2)



February 2018, Ankush Chadha gives a talk at Helm Summit on index.yaml scalability concerns

- ▶ Later that year, Matt Farina submits an in-depth proposal for index partitioning in Helm 3

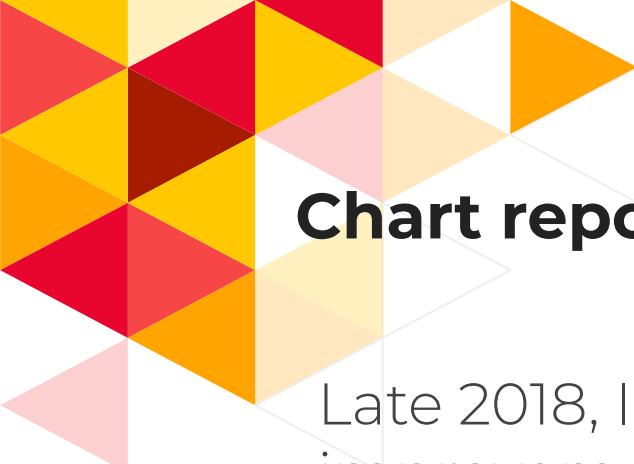


Chart repos fall short (pt. 3)

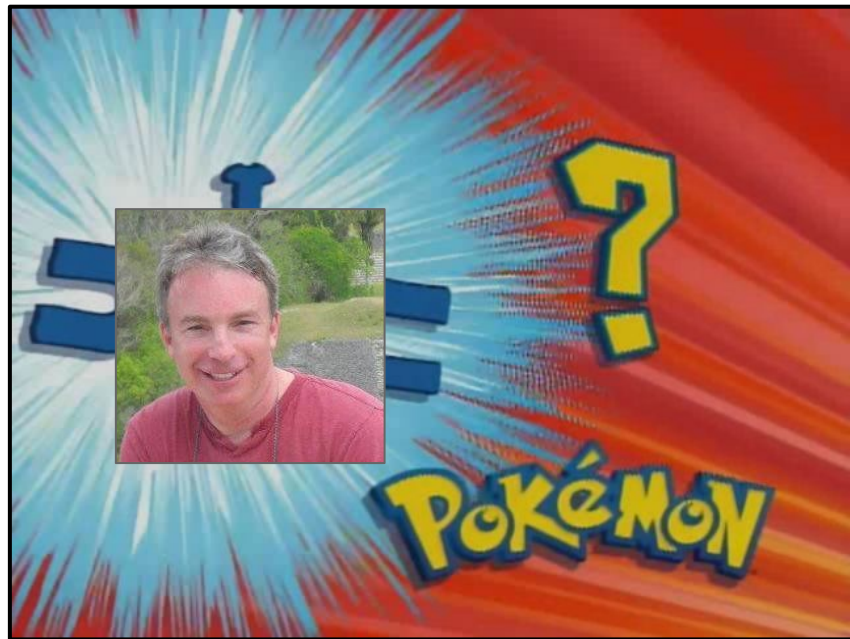
Late 2018, I begin hosting weekly meetings to plan improvements for Helm 3 repo system

- ▶ How do we enable fine-grained authorization?
- ▶ How do we enable chart uploads?
- ▶ How do we “helm login” ?
- ▶ How do we “helm push” ?

And then...



a wild Steve Lasker appears!



Steve: Helm should work like Docker!

 [helm](#) / [community](#)

Unwatch ▾ 36

★ Star 118

Fork 58

[Code](#) [Issues 8](#) [Pull requests 9](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

Proposal to converge helm repositories with docker registries #55

Edit

Open

 SteveLasker wants to merge 4 commits into `helm:master` from `AzureCR:convergence`

Conversation 24

Commits 4

Checks 1

Files changed 10

+319 -0



SteveLasker commented on Oct 26, 2018 • edited ▾

First-time contributor + 😊 ...

@jdolitsky discussion on convergence of helm repos and container registries for Helm 3

helm-bot added the `size/L` label on Oct 26, 2018

Proposal to converge helm repositories with docker registries ...

✓ 6661a67

Reviewers

 jdolitsky

 ant31

 eldada

Assignees

No one—assign yourself

**This all led to
“seemingly interminable
registry discussions”...**





Then - just in time for KubeCon NA 2018...

Helm team decides that somehow, somehow, the next generation of chart distribution will be powered by OCI

- ◀ Conversations moved to OCI call/mailing list
- ◀ Still leaves open many, many questions
- ◀ “What’s going to happen to the museum?”
- ◀ Also.. OC-what???

OCI = Open Container Initiative



OPEN CONTAINER
INITIATIVE



OPEN CONTAINER INITIATIVE

AN OPEN GOVERNANCE STRUCTURE FOR THE EXPRESS PURPOSE
OF CREATING OPEN INDUSTRY STANDARDS AROUND CONTAINER
FORMATS AND RUNTIME

Super Brief History of OCI (pt. 1)

2013

Docker is released



2014

rkt is released (CoreOS)



2015

OCI launched as *the* open container runtime+image spec with support from Docker, CoreOS, Google, RedHat, etc.

Super Brief History of OCI (pt. 2)



Fast-forward a bit...

2018

OCI adds **distribution spec** ← ?



The OCI Distribution Specification

- ▶ Announced in April 2018
- ▶ A standard for container image distribution
- ▶ Based on the Docker Registry HTTP API (v2)
- ▶ Supports the pushing and pulling of container images



```
graph TD; A[docker push] --> B[Azure Registry]; A --> C[QUAY]; A --> D[AWS ECR]; A --> E[Google Container Registry]; A --> F[dockerhub]; A --> G[HARBOR];
```

docker push



Azure Registry



AWS ECR



Google Container Registry



HARBOR



dockerhub



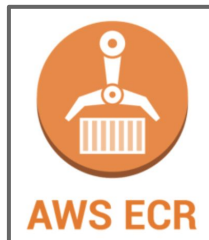
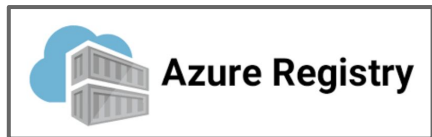
@jdolitsky



```
docker push
```

`POST /v2/<name>/blobs/uploads/`

 This is the OCI Distribution Spec!!



@jdolitsky



Leveraging the OCI Distribution Spec for other things (such as Helm charts)

OCI = Open *Containers* Initiative - but the distribution spec provides value as a generalized API for storage

- ◀ So now what?

Enter the ORAS project:

ORAS

OCI Registry As Storage



The ORAS project: a toolset for multi-artifact OCI registry support in Go

- ▶ Microsoft engineer Shiwei Zhang creates the first version of ORAS based on existing work he and his team has done for storing Helm charts in Azure Container Registry
- ▶ I help ensure that the library works with Helm
- ▶ Allows pushing of custom artifacts to an OCI registry, and specifying different mediatypes



github.com/deislabs/oras



@jdolitsky

Helm 3 uses the ORAS library - so you can now do things like this:

```
$ export HELM_EXPERIMENTAL_OCI=1
```



Required for now

```
$ helm registry login mysite.io
```

```
$ helm chart pull mysite.io/somechart:1.5.0
```

```
$ helm chart save mychart/ mysite.io/mychart:2.0.1
```

```
$ helm chart push mysite.io/mychart:2.0.1
```

For more info on Helm 3 registry support:

→ **v3.helm.sh/docs/topics/registries** ←

Also - join the workshop at 14:35 today!

**So now we can push Helm charts,
but where do we push them to??**



That's a bit of a tricky question...

Support for new artifact types are limited on public hubs/registries due to reasons such as UX, security, etc.



Two (2) primary open source options to deploy our own registry:

- 1.
- 2.

Two (2) primary open source options to deploy our own registry:

1. Docker Distribution
- 2.



Docker Distribution: The Docker toolset to pack, ship, store, and deliver content

- ▶ Also known as “Docker Registry”
- ▶ Adheres to the OCI spec (because the spec was based upon it!)
- ▶ Several config options, storage backends, etc.
- ▶ Relatively easy to get started using Docker
- ▶ **\$ docker run --rm -p 5000:5000 registry**



github.com/docker/distribution

@jdolitsky

Built on top of Docker Distribution:

Bundle Bar

A registry for lightweight, cloud-native artifacts



bundle.bar

bundle.bar

Welcome to

Bundle Bar!



Bundle Bar is a free service that hosts new types of lightweight, cloud-native artifacts for use by tools leveraging [OCI](#) for package distribution.

Community-provided, public artifacts are made available for download. Please see below for a complete list of artifacts hosted by type.

Supported Artifact Types

Helm Charts

[Helm 3](#) provides support for downloading chart packages from OCI-based registries. For more info on how to use these features, please see [this doc](#).

Here's an example of how to install one of the charts below using Helm 3:

```
export HELM_EXPERIMENTAL_OCI=1
helm chart pull bundle.bar/helm/hub/bitnami/nginx:4.3.3
helm chart export bundle.bar/helm/hub/bitnami/nginx:4.3.3
helm install myrelease nginx/
```

The following chart packages have been imported from [Helm Hub](#) (880 total):

```
bundle.bar/helm/hub/agones/agones:0.12.0
bundle.bar/helm/hub/appdynamics/cluster-agent:0.1.1
bundle.bar/helm/hub/appdynamics/machine-agent:0.2.3
bundle.bar/helm/hub/appcode/csi-vault:0.2.0
bundle.bar/helm/hub/appcode/g2:0.3.2
```

But what about..

- Vulnerability scanning?
- Content trust?
- SSO integration and RBAC?
- Replication?
- User interface?

Two (2) primary open source options to deploy our own registry:

1. Docker Distribution
- 2. Harbor!!**



Harbor: an open source cloud native registry that stores, signs, and scans container images for vulnerabilities

- ▶ Originally a VMware project, now part of CNCF
- ▶ Adds extra features on top of Docker Distribution
- ▶ Nice graphical UI to manage images, permissions
- ▶ OpenID Connect (OIDC) and LDAP/AD support
- ▶ Collection of tools such as Notary, Claire, etc.
- ▶ Actually already supports charts via ChartMuseum



github.com/goharbor/harbor

@jdolitsky



Deploying Harbor with Helm 2

Easy right?

```
$ helm repo add harbor https://helm.goharbor.io  
$ helm install --name=harbor harbor/harbor
```



github.com/goharbor/harbor



@jdolitsky

Deploying Harbor with Helm 2

```
> kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
harbor-harbor-chartmuseum-bf97b6df6-dwktq	1/1	Running	0	109s
harbor-harbor-clair-5bd8f76d9c-q9sm4	1/1	Running	0	109s
harbor-harbor-core-855fc68cdd-7rgrn	1/1	Running	0	108s
harbor-harbor-database-0	1/1	Running	0	107s
harbor-harbor-jobservice-6c58df48c5-vtp7f	1/1	Running	0	108s
harbor-harbor-notary-server-6f6bb7874-ptcnz	1/1	Running	1	108s
harbor-harbor-notary-signer-59988f796c-2qg42	1/1	Running	1	108s
harbor-harbor-portal-64cff84747-fvnbg	1/1	Running	0	108s
harbor-harbor-redis-0	1/1	Running	0	107s
harbor-harbor-registry-56cd7b9c79-rpjks	2/2	Running	0	108s



github.com/goharbor/harbor

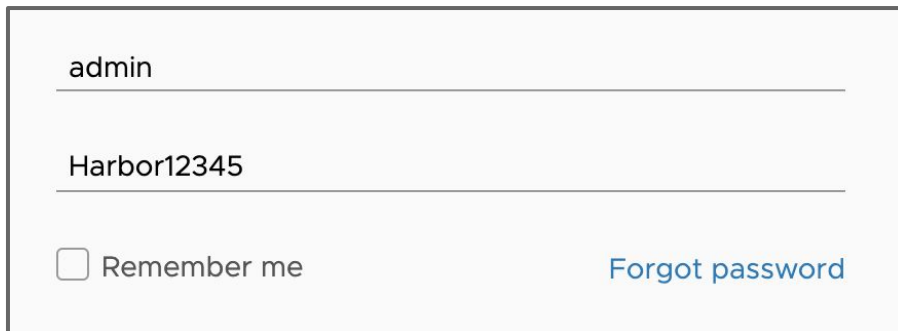
@jdolitsky



Accessing the Harbor UI

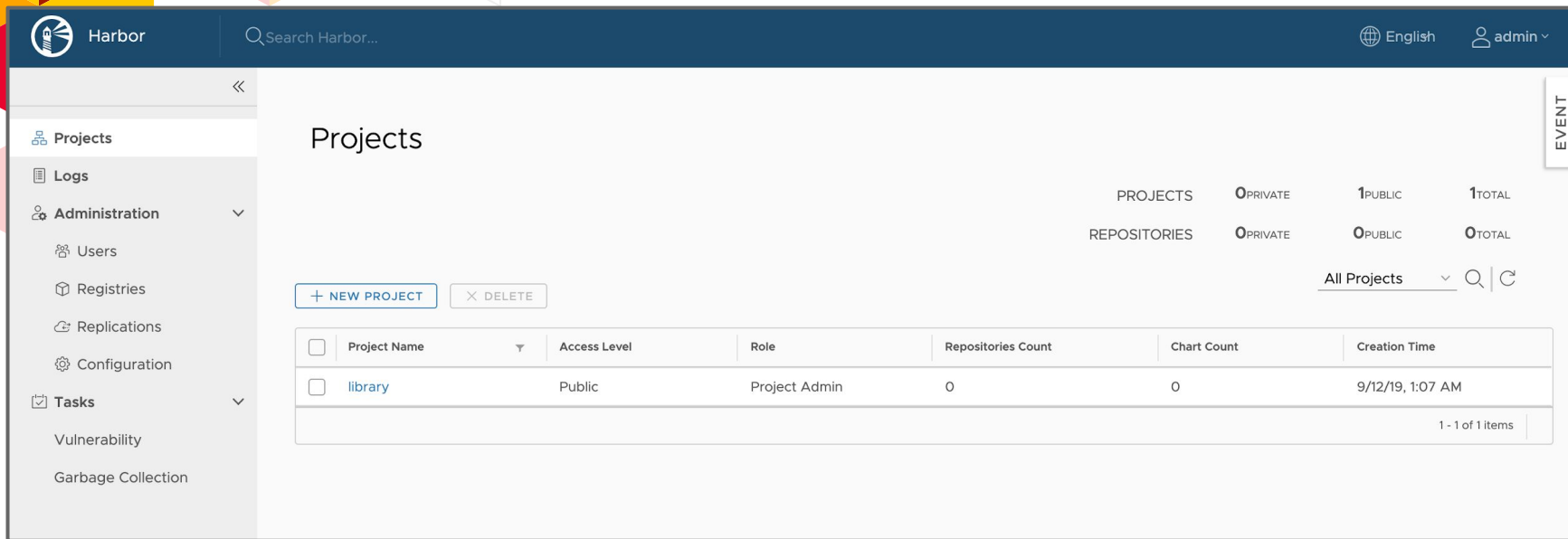
```
$ POD_NAME="$(kubectl get pods \
  | grep core | awk '{print $1}')"
$ kubectl port-forward $POD_NAME 8080:8080
```

Go to <http://localhost:8080> in your browser,
and enter default admin credentials:



A screenshot of the Harbor login interface. It features two input fields: the first is labeled 'admin' and the second is labeled 'Harbor12345'. Below these fields is a checkbox labeled 'Remember me'. To the right of the checkbox is a blue link that says 'Forgot password'.

Harbor UI



A screenshot of the Harbor UI interface. The top navigation bar is dark blue with the Harbor logo, a search bar, and language/user settings. The left sidebar contains a menu with 'Projects' selected. The main content area shows the 'Projects' page with a summary of project statistics and a table of existing projects.

Harbor Search Harbor... English admin

Projects

PROJECTS 0 PRIVATE 1 PUBLIC 1 TOTAL
REPOSITORIES 0 PRIVATE 0 PUBLIC 0 TOTAL

All Projects

+ NEW PROJECT X DELETE

☐	Project Name	Access Level	Role	Repositories Count	Chart Count	Creation Time
☐	library	Public	Project Admin	0	0	9/12/19, 1:07 AM

1 - 1 of 1 items

Tweaking Harbor to accept Helm charts

```
# helm-support.yaml
core:
  image:
    repository: bloodorangeio/harbor-core
    tag: master
```

OCI Artifacts support is still in-progress in Harbor,
so we use a fork with a few small workarounds

Tweaking Harbor to accept Helm charts

```
# helm-support.yaml
core:
  image:
    repository: bloodorangeio/harbor-core
    tag: master

notary:
  enabled: false
externalURL: https://core.harbor.bundle.bar
expose:
  ingress:
    hosts:
      core: core.harbor.bundle.bar
    annotations:
      kubernetes.io/ingress.class: nginx
      kubernetes.io/tls-acme: "true"
      ingress.kubernetes.io/proxy-body-size: "1m"
```

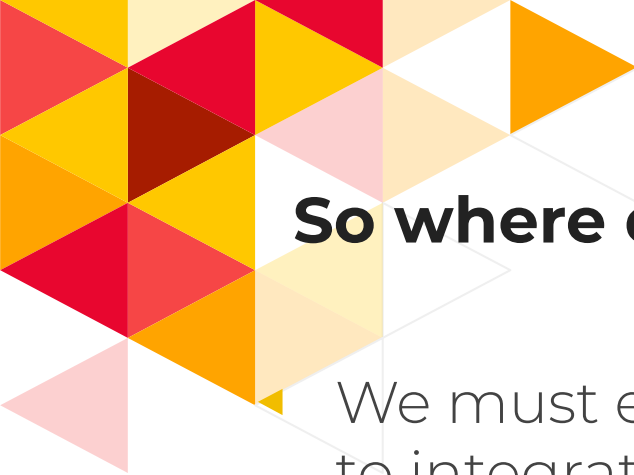
Disable notary (for now), enable ingress, request TLS cert via cert-manager, allow 1MB uploads

Tweaking Harbor to accept Helm charts

```
$ helm upgrade --install \  
  harbor harbor/harbor \  
  -f helm-support.yaml
```

Pushing charts to Harbor with Helm 3

```
[jdolitsky@uno:~/hs] $ helm fetch harbor/harbor
[jdolitsky@uno:~/hs] $ ls *.tgz
harbor-1.1.2.tgz
[jdolitsky@uno:~/hs] $ export HELM_EXPERIMENTAL_OCI=1
[jdolitsky@uno:~/hs] $ helm chart save harbor-1.1.2.tgz helmsummit.bundle.bar/library/harbor
ref:      helmsummit.bundle.bar/library/harbor:1.1.2
digest:   0f8c0650d55f5e00d11d7462381c340454a3b9e517e15a0187011dc305690541
size:     28.1 KiB
name:     harbor
version:  1.1.2
1.1.2: saved
[jdolitsky@uno:~/hs] $ helm registry login -u admin -p Harbor12345 helmsummit.bundle.bar
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
Login succeeded
[jdolitsky@uno:~/hs] $ helm chart push helmsummit.bundle.bar/library/harbor:1.1.2
The push refers to repository [helmsummit.bundle.bar/library/harbor]
ref:      helmsummit.bundle.bar/library/harbor:1.1.2
digest:   0f8c0650d55f5e00d11d7462381c340454a3b9e517e15a0187011dc305690541
size:     28.1 KiB
name:     harbor
version:  1.1.2
1.1.2: pushed to remote (1 layer, 28.1 KiB total)
[jdolitsky@uno:~/hs] $
```



So where does Helm stand now?

We must evolve the experimental Helm 3 OCI work to integrate with the rest of Helm UX. For example:

- ▶ **\$ helm install mysite.io/mychart:2.0.1**
- ▶ We must follow/engage in the conversation going on in the new “OCI Artifacts” project:
 - ▶ **github.com/opencontainers/artifacts**

Most importantly, we need YOU!

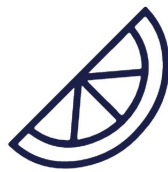
- Use it
- Break it
- Fix it
- Live it

Sources

- **"Quay: Introducing an Application Registry for Kubernetes"** - Antoine Legrand, 2017-03-23
<https://coreos.com/blog/quay-application-registry-for-kubernetes.html>
- **"Helm: Understanding Charts API vs App Registry"** - Matt Farina, 2017-09-25
<https://codeengineered.com/blog/2017/helm-charts-api-vs-app-registry/>
- **"The Hitchhiker's Guide to the Container Galaxy"** - Itay Shakury, 2018-02-06
<https://blog.itaysk.com/2018/02/06/the-hitchhickers-guide-to-the-container-galaxy>
- **"Cloud Native Artifact Registries evolve from Docker Container Registries"** - Steve Lasker, 2019-01-25
<https://stevelasker.blog/2019/01/25/cloud-native-artifact-stores-evolve-from-container-registries/>
- **"Distributing with Distribution: Upcoming Changes to Helm Chart Repositories"** - Matt Fisher, 2019-01-25
<https://blog.bacongobbler.com/post/2019-01-25-distributing-with-distribution/index.html>
- **"OCI Artifacts"** - Jimmy Zelinskie, 2019-08-23
<https://medium.com/@jzelinskie/oci-artifacts-f5f3b4ef2889>

Thank you!



 **bloodorange.io**