

OpenShift

Descargar programa

Para descargarlo primero hay que registrarse en la pagina de redhat, seguir los pasos y luego descargar el programa

The screenshot shows the Red Hat Developer website's product section. It includes a navigation bar with links to Products, Technologies, Learn, Developer Sandbox, Blog, Events, and Videos. Below the navigation, there are two columns: 'Platforms' and 'Featured'. The 'Platforms' column lists Red Hat Enterprise Linux, Red Hat AI, Red Hat OpenShift, and Red Hat Ansible Automation Platform. The 'Featured' column lists Red Hat build of OpenJDK, Red Hat Developer Hub, Red Hat JBoss Enterprise Application Platform, Red Hat OpenShift Dev Spaces, and Red Hat OpenShift Local. A red underline highlights the 'Red Hat OpenShift Local' link. At the bottom of the section is a link to 'View All Red Hat Products →'.

Boton rojo

The screenshot shows the Red Hat OpenShift Local landing page. The title is 'Red Hat OpenShift Local (formerly Red Hat CodeReady Containers)'. Below the title, it says 'OpenShift Local on your laptop gets you up and running with an OpenShift cluster on your local machine in minutes.' There are two buttons: 'Install OpenShift on your laptop' (in red) and 'Try OpenShift in our free Developer Sandbox'.

Descargar archivo

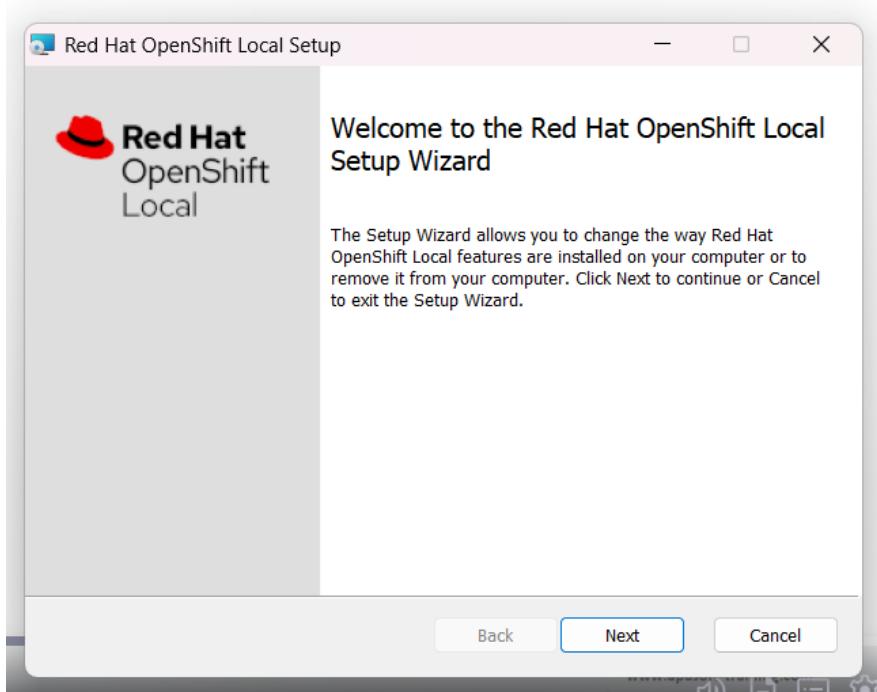
The screenshot shows the 'Download what you need to get started' section for OpenShift Local. It includes a note about extracting the archive and opening the installer, which will start an installation guide. It features dropdown menus for 'Windows' and 'x86_64' operating systems, and a 'Download OpenShift Local' button. Below this, the 'Pull secret' section is shown with 'Download pull secret' and 'Copy pull secret' buttons.

Instalar el programa

The screenshot shows a file manager interface with a table of files. The table has columns for Nombre (Name), Tipo (Type), Tamaño comprimido (Compressed Size), Protegido ... (Protected), Tamaño (Size), Relación (Relationship), and Fecha de m... (Last modified). One file is listed: 'crc-windows-amd64.msi' with a size of 41,787 KB and a compressed size of 42,500 KB. The last modified date is 6/10/2025.

Nombre	Tipo	Tamaño comprimido	Protegido ...	Tamaño	Relación	Fecha de m...
crc-windows-amd64.msi	Paquete de Windows Insta...	41,787 KB	No	42,500 KB	2%	6/10/2025

Seguir la instalacion con next, next, ...



Configurar el setup en modo administrador (esto puede demorar mas de 1h)

```
crc setup
```

Iniciar openshift sin administrador (demora unos 10 min)

```
crc start
```

Credenciales para el servicio (importante: copiar esta informacion y guardarla)

The server is accessible via web console at:
<https://console-openshift-console.apps-crc.testing>

Log in as administrator:
Username: kubeadmin
Password: au5kW-d7V8B-EHvZr-MUwct

Log in as user:
Username: developer
Password: developer

Use the 'oc' command line interface:
PS> & crc oc-env | Invoke-Expression
PS> oc login -u developer https://api.crc.testing:6443

Aceptar el modo inseguro del navegador

The screenshot shows the Red Hat OpenShift Local cluster interface. The main header says "OpenShift Local cluster is for development and testing purposes. DON'T use it for production." The left sidebar has links like Inicio, Favoritos, Operadores, Helm, Cargas de trabajo, Redes, Almacenamiento, Compilaciones, Calcular, Gestión de usuarios, and Administración. The main content area is titled "Descripción general" and has tabs for Clúster, Servicios, and Recursos. Under Clúster, there's a "Recursos para empezar" section with "Configurar su clúster" and "Realizar recorrido por la consola". There's also a "Desarrollar con documentación guida" section with "Enable the Developer Perspective" and "Impersonating the system:admin user". A sidebar on the right says "Explora nuevas funciones y capacidades" with links to "OpenShift AI" and "OpenShift Lightspeed". The "Estado" section shows the cluster is "Operador" and the "Actividad" section shows "En curso" with "No hay actividades en curso".

Comandos basicos

```
crc status  
crc stop  
crc delete  
crc start  
crc console  
crc config
```

Instalar cliente OpenShift

The screenshot shows the Red Hat Hybrid Cloud Console interface. The left sidebar has links for Services, OpenShift (selected), Clusters, Overview, Releases, Developer Sandbox, Downloads (selected), Advisor, Vulnerability Dashboard, and Subscriptions. The main content area is titled "Downloads" and shows "Command-line interface (CLI) tools". It lists "OpenShift command-line interface (oc)" and "OpenShift Cluster Manager API command-line Interface (ocm)". For each, there are dropdown menus for "OS type" (Windows or Linux) and "Architecture type" (x86_64 or x64). There are also "Download" buttons for each item.

Otra opcion

The screenshot shows the Red Hat OpenShift Local cluster interface. The top right has a dropdown menu for "kubeadmin". The menu items are "Quick Starts", "Documentation", "Command Line Tools" (which is highlighted with a blue background), and "Guided tour".

Crear token de acceso

Es necesario loguearse con un usuario puder como kubeadmin o developer

Your API token is
sha256~0EIX0xNl9Q_04-iQPRUGf4989OXmmD4TTOL7jDJ6CqE

Log in with this token

```
oc login --token=sha256~0EIX0xNl9Q_04-iQPRUGf4989OXmmD4TTOL7jDJ6CqE --server=https://api.crc.testing:6443
```

Use this token directly against the API

```
curl -H "Authorization: Bearer sha256~0EIX0xNl9Q_04-iQPRUGf4989OXmmD4TTOL7jDJ6CqE" "https://api.crc.testing:6443/apis/user.openshift.io/v1/users/~"
```

[Request another token](#)

```
oc login --token=sha256~0EIX0xNl9Q_04-iQPRUGf4989OXmmD4TTOL7jDJ6CqE --server=https://api.crc.testing:6443
```

Proyectos

Los proyectos es lo mismos que los namespaces de kubernetes pero con mas funcionalidades como nuevos objetos de openshift

```
oc new-project dev1
```

```
PS C:\Users\g10n_> oc new-project dev1
Now using project "dev1" on server "https://api.crc.testing:6443".
You can add applications to this project with the 'new-app' command. For example, try:
  oc new-app rails-postgresql-example
to build a new example application in Ruby. Or use kubectl to deploy a simple Kubernetes application:
  kubectl create deployment hello-node --image=registry.k8s.io/e2e-test-images/agnhost:2.43 -- /agnhost serve-hostname
```

```
oc get project
```

```
PS C:\Users\g10n_> oc get project
NAME          DISPLAY NAME   STATUS
default        Active
dev1          Active
hostpath-provisioner Active
kube-node-lease Active
kube-public    Active
kube-system    Active
openshift      Active
openshift-apiserver Active
openshift-apiserver-operator Active
openshift-authentication Active
```

Setear namespace/project

```
oc config set-context --current --namespace=dev1
```

Configurar cliente OC

```
[Environment]::SetEnvironmentVariable("PATH", $env:PATH + ";C:\Users\g10n_\oc",
[EnvironmentVariableTarget]::User)
$env:PATH = [Environment]::GetEnvironmentVariable("PATH", [EnvironmentVariableTarget]::User)
if (-not ($env:PATH -like '*\oc*')) { $env:PATH += ';C:\Users\g10n_\oc'}
oc version
```

Configurar cliente OC en WSL (Linux)

```
# Actualizar el sistema WSL
sudo apt update && sudo apt upgrade -y

# Instalar curl y wget si no están instalados
sudo apt install curl wget -y

# Descargar OpenShift CLI (oc) para Linux
wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/stable/openshift-client-linux.tar.gz

# Extraer el archivo descargado
tar -xzf openshift-client-linux.tar.gz

# Crear directorio para binarios locales si no existe
mkdir -p ~/.local/bin

# Mover el binario oc al directorio local
mv oc ~/.local/bin/

# Hacer el binario ejecutable
chmod +x ~/.local/bin/oc

# Agregar el directorio al PATH en .bashrc
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc

# Recargar el archivo .bashrc
source ~/.bashrc

# Verificar la instalación de oc
oc version --client

# Obtener la IP de Windows desde WSL
ip route show | grep -i default | awk '{ print $3}'

# Agregar la IP de Windows al archivo hosts (reemplaza 172.23.208.1 con tu IP real)
echo "172.23.208.1 api.crc.testing" | sudo tee -a /etc/hosts

# Verificar que la resolución DNS funciona correctamente
getent hosts api.crc.testing

# Configurar un nuevo cluster apuntando al proxy de Windows
oc config set-cluster crc-wsl --server=http://172.23.208.1:8080 --insecure-skip-tls-verify=true

# Crear un nuevo contexto que use el cluster configurado
oc config set-context crc-wsl --cluster=crc-wsl --user=kubeadmin/api-crc-testing:6443 --namespace=default

# Cambiar al nuevo contexto
oc config use-context crc-wsl

# Verificar la configuración actual
oc config current-context

# Probar la conectividad básica
oc get pods

# Probar conectividad con todos los namespaces
oc get pods --all-namespaces

# Verificar los nodos del cluster
oc get nodes

# Limpiar archivos temporales
rm -f openshift-client-linux.tar.gz kubectl
```

YAML kubernetes/OpenShift

Sigue la misma estructura que Kubernetes con los 4 objetos importantes: api, kind, metadata, spec.

```
apiVersion: project.openshift.io/v1
kind: Project
metadata:
  annotations:
    openshift.io/description: Esto es la descripción del proyecto
    openshift.io/display-name: Ejemplo de creación de un proyecto OpenShift
    openshift.io/requester: developer
    documentacion: Ejemplo para crear un proyecto en OpenShift
  name: desa2
  labels:
    tipo: desa
spec:
  finalizers:
    - kubernetes
```

Obtener el yaml de un objeto

```
oc get project dev1 -o yaml
```

Ver proyecto desde la web

The screenshot shows the OpenShift web interface. On the left, there's a sidebar with navigation links like Home, Overview, Projects (which is selected), Search, Software Catalog, API Explorer, Events, Favorites, Operators, and Helm. The main area is titled 'Projects' and lists several projects:

Name	Display name	Status	Requester	Memory	CPU	Created
PR default	No display name	Active	No requester	-	-	Sep 24, 2025, 9:20 AM
PR dev1	No display name	Active	kubeadmin	-	-	Oct 11, 2025, 2:48 PM
PR hostpath-provisioner	No display name	Active	No requester	-	-	Sep 25, 2025, 10:23 AM
PR kube-node-lease	No display name	Active	No requester	-	-	Sep 24, 2025, 9:20 AM

Below this, a specific project detail page is shown for 'PR dev1'. The top bar shows 'PR dev1 Active'. The page has tabs for Overview, Details, YAML, Workloads, and RoleBindings. The Overview section contains a 'Getting started resources' section with three items: 'Create applications using samples', 'Build with guided documentation', and 'Explore new developer features'.

```

1 kind: Project
2 apiVersion: project.openshift.io/v1
3 metadata:
4   name: dev1
5   uid: f0fc29ac-0349-4a06-9f36-0c8652382128
6   resourceVersion: '52236'
7   creationTimestamp: '2025-10-11T13:48:26Z'
8   labels:
9     kubernetes.io/metadata.name: dev1
10    pod-security.kubernetes.io/audit: restricted
11    pod-security.kubernetes.io/audit-version: latest
12    pod-security.kubernetes.io/warn: restricted
13    pod-security.kubernetes.io/warn-version: latest
14   annotations:
15     openshift.io/description: ''
16     openshift.io/display-name: ''

```

Save Reload Cancel Download

Crear proyecto con yaml

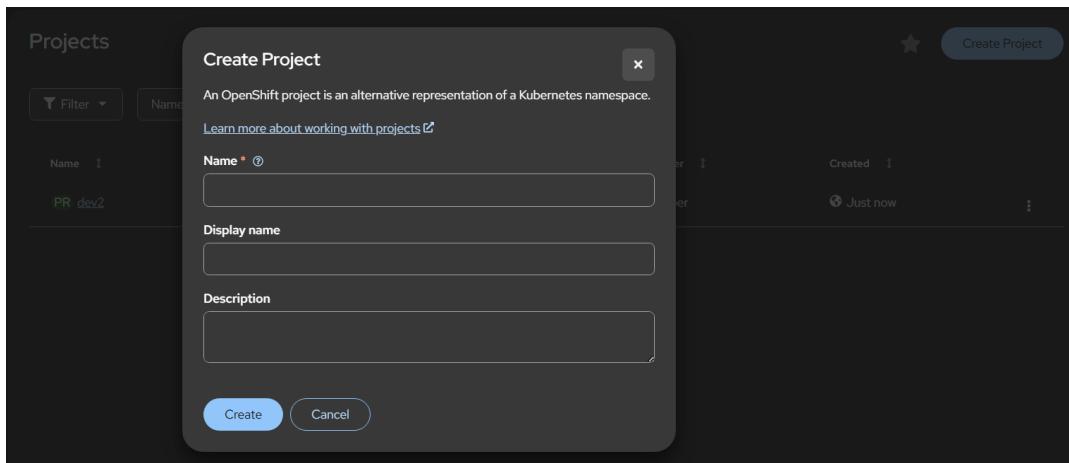
```

apiVersion: project.openshift.io/v1
kind: Project
metadata:
  annotations:
    openshift.io/description: Esto es la descripcion del proyecto
    openshift.io/display-name: Ejemplo de creacion de una proyecto Openshift
    openshift.io/requester: developer
    documentacion: Ejemplo para crear un proyecto en openshift
  name: desa2
  labels:
    tipo: desa
spec:
  finalizers:
  - kubernetes

```

```
oc apply -f proyecto.yaml
```

Desde la web



```

PS C:\Users\g10n_> oc get project dev2
NAME      DISPLAY NAME      STATUS
dev2      Proyecto 2      Active
PS C:\Users\g10n_>

```

IMPORTANTE: si creo un namespace en openshift crea un project y viceversa

Seguridad en Openshift *

Configurar permisos para poder evitar errores de seguridad.

Algunas imágenes que usaremos durante el curso requiere privilegios de acceso como ROOT o bien necesitan ciertos permisos para acceder a volúmenes o puertos.

Por ejemplo postgres, redis, Apache, etc.

Aunque este curso no es de Administración, necesitamos dar ciertos permisos al usuario para que pueda trabajar.

Es necesario ejecutar el siguiente comando en cada uno de los proyectos que creemos durante el curso

```
oc adm policy add-scc-to-user anyuid -z default
```

De esa forma podremos crear objetos y contenedores sin problemas.

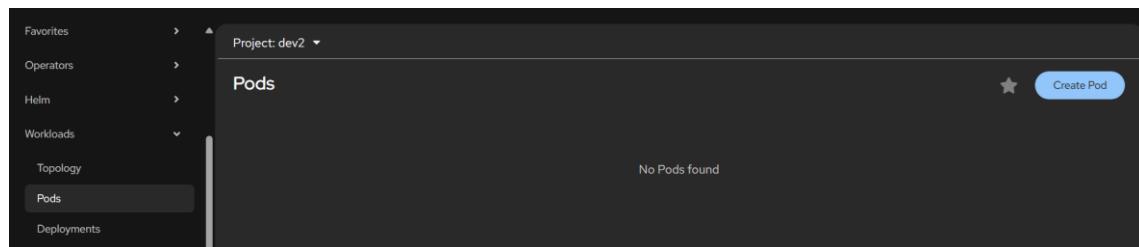
```
PS C:\Users\g10n_> oc adm policy add-scc-to-user anyuid -z default
clusterrole.rbac.authorization.k8s.io/system:openshift:scc:anyuid added: "default"
PS C:\Users\g10n_>
```

Crear pod

```
oc run nginx1 --image=nginx
oc get pod
oc describe pod nginx1
```

```
PS C:\Users\g10n_> oc run nginx1 --image=nginx
pod/nginx1 created
PS C:\Users\g10n_> oc get pod
NAME      READY   STATUS    RESTARTS   AGE
nginx1   0/1     ContainerCreating   0          4s
PS C:\Users\g10n_> oc describe pod nginx1
Name:           nginx1
Namespace:      dev1
Priority:       0
Service Account: default
Node:           crc/192.168.126.11
Start Time:     Sat, 11 Oct 2025 15:31:40 +0100
Labels:          run=nginx1
Annotations:    k8s.ovn.org/pod-networks:
                  {"default":{"ip_addresses":["10.217.0.80/23"],"mac_address":"0a:58:0a:d9:00:50","gateway_17.0.1"},"routes":[{"dest":"10.217.0.0..."}]
k8s.v1.cni.cncf.io/network-status:
[{
      "name": "ovn-kubernetes",
      "interface": "eth0",
      "ips": [
        "10.217.0.80"
      ],
    },
```

En la web



Project: devl

Pod details

Pod nginx1 (Running)

Details Metrics YAML Environment Logs Events Terminal

Pod details

Name: nginx1 Status: Running

Namespace: NS devl Restart policy: Always restart

Labels: run=nginx1 Active deadline seconds: Not configured

Node selector: Pod IP:

Logs

Pods > Pod details

Pod nginx1 (Running)

Details Metrics YAML Environment Logs Events Terminal

Log streaming... Current log Search

Show full log Wrap lines Raw Download Expand

```
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
5 /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
6 /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
7 /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
8 /docker-entrypoint.sh: Configuration complete; ready for start up
9
10 /docker-entrypoint.sh: 2025/10/11 14:31:46 [notice] 1#1: using the "epoll" event method
11 2025/10/11 14:31:46 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
12 2025/10/11 14:31:46 [notice] 1#1: OS: Linux 5.14.0-570.45.1.e19.6.x86_64
13 2025/10/11 14:31:46 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
14 2025/10/11 14:31:46 [notice] 1#1: start worker processes
15 2025/10/11 14:31:46 [notice] 1#1: start worker process 24
16 2025/10/11 14:31:46 [notice] 1#1: start worker process 25
17 2025/10/11 14:31:46 [notice] 1#1: start worker process 26
18 2025/10/11 14:31:46 [notice] 1#1: start worker process 27
19 2025/10/11 14:31:46 [notice] 1#1: start worker process 28
```

Se puede usar una terminal para ejecutar desde el pod

Project: devl

Pods > Pod details

Pod nginx1 (Running)

Details Metrics YAML Environment Logs Events Terminal

Connecting to nginx1

```
# pwd
/
# ls
bin  dev  docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot docker-entrypoint.d  etc          lib   media  opt  root  sbin  sys  usr
#
```

Editar algunas opciones del pod

Project: devl

Pod details

Pod nginx1 (Running)

Details Metrics YAML Environment Logs Events Terminal

Connecting to nginx1

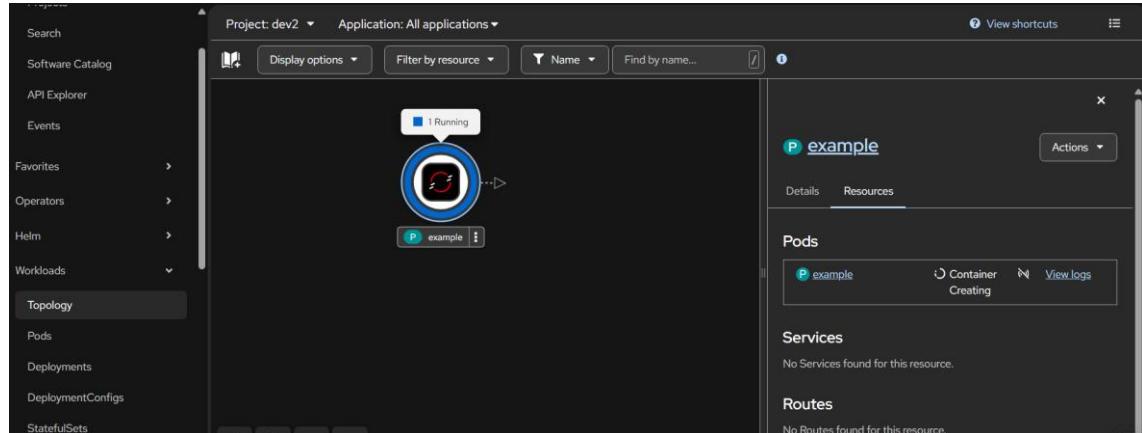
```
# pwd
/
```

Actions

- Edit labels
- Edit annotations
- Edit Pod
- Delete Pod

Pods en modo desarrollo

Muestra una sección de topology, donde podemos ver el estado del pod



Pod en la web

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx2
  labels:
    app: nginx-app
  namespace: desal
spec:
  containers:
    - name: nginx-app
      image: nginx
      ports:
        - containerPort: 80
```

Crear objetos de yaml desde la web

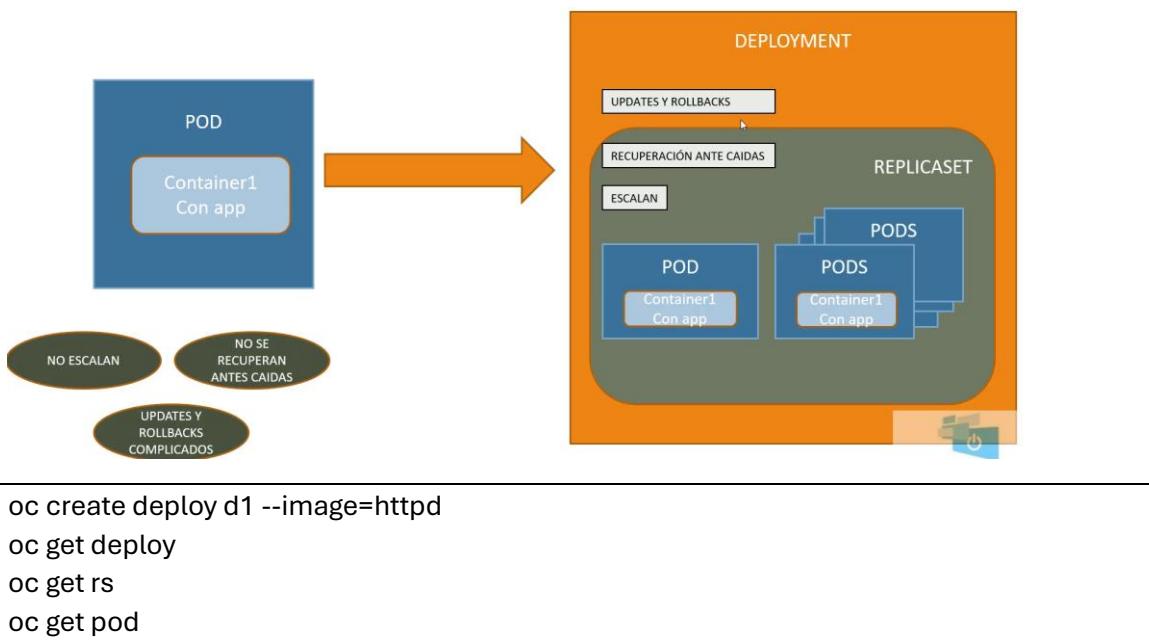
Import YAML

Drag and drop YAML or JSON files into the editor, or manually enter files and use `---` to separate each definition.

```
1
```

Deployment

Openshift desde cero

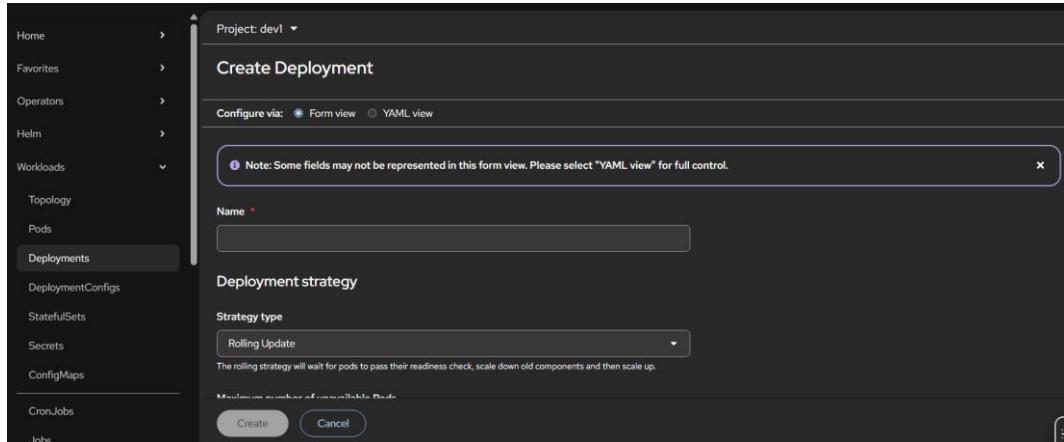


```
PS C:\Users\g10n_> oc create deploy d1 --image=httpd
deployment.apps/d1 created
PS C:\Users\g10n_> oc get deploy
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
d1        1/1     1            1           8s
PS C:\Users\g10n_> oc get svc
No resources found in devl namespace.
PS C:\Users\g10n_> oc get rs
NAME      DESIRED   CURRENT   READY   AGE
d1-7956fbf4db  1         1         1       29s
PS C:\Users\g10n_> oc get pod
NAME      READY   STATUS    RESTARTS   AGE
d1-7956fbf4db-74kp5  1/1     Running   0          34s
nginxx1   1/1     Running   0          86m
PS C:\Users\g10n_>
```

A screenshot of the OpenShift web interface. The top navigation bar shows 'Project: devl'. Below it, a breadcrumb navigation shows 'Deployments > Deployment details'. The main area displays a deployment named 'd1'. The 'ReplicaSets' tab is selected. A table lists one replica set: 'RS d1-7956fbf4db' with '1 of 1 pods' status, owned by 'd1', and created 2 minutes ago. Other tabs include 'Details', 'Metrics', 'YAML', 'Pods', 'Environment', and 'Events'.

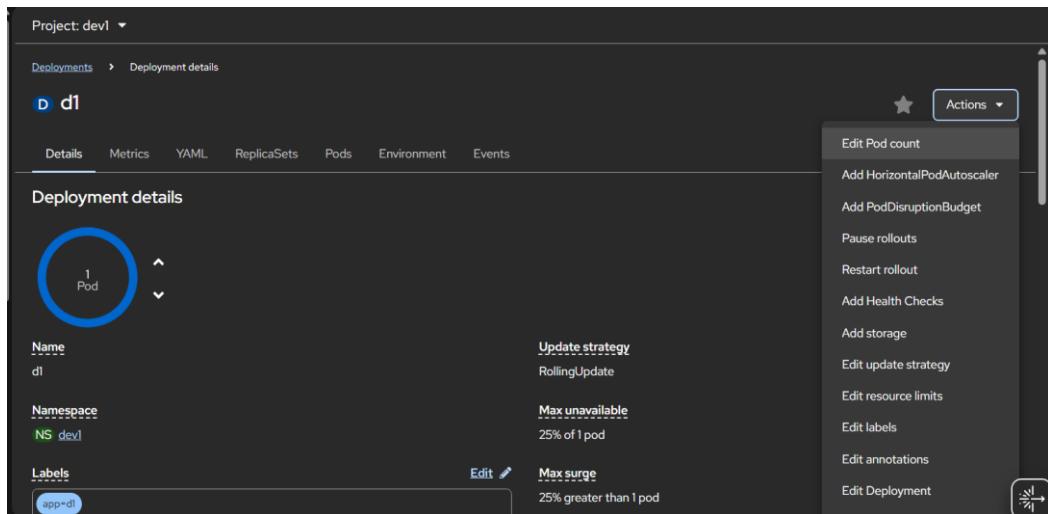
Crear deployment en la web

Se puede crear de 2 formas: del tipo formulario y del tipo yaml



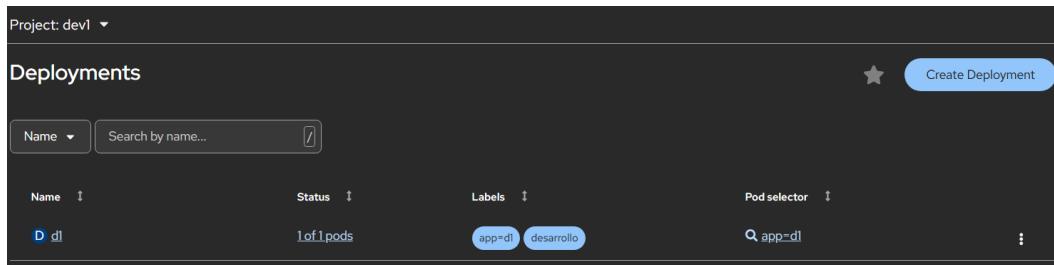
The screenshot shows the 'Create Deployment' interface in the Kubernetes UI. On the left, a sidebar lists various workload types: Home, Favorites, Operators, Helm, Workloads (with Deployments highlighted), Topology, Pods, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, and Jobs. The main area is titled 'Create Deployment' and includes a note: 'Note: Some fields may not be represented in this form view. Please select "YAML view" for full control.' It has sections for 'Name' (set to 'd1'), 'Deployment strategy' (strategy type is 'Rolling Update'), and a note about the rolling strategy. At the bottom are 'Create' and 'Cancel' buttons.

Acciones de los deployments



The screenshot shows the 'Deployment details' page for 'd1'. It displays basic information: 1 Pod, Name: d1, Namespace: NS dev, Labels: app=d1. It also shows the 'Update strategy': RollingUpdate, Max unavailable: 25% of 1 pod, and Max surge: 25% greater than 1 pod. A context menu is open on the right, listing actions such as Edit Pod count, Add HorizontalPodAutoscaler, Add PodDisruptionBudget, Pause rollouts, Restart rollout, Add Health Checks, Add storage, Edit update strategy, Edit resource limits, Edit labels, Edit annotations, and Edit Deployment.

Aplicar cambios en deployments



The screenshot shows the 'Deployments' list page. It lists one deployment named 'd1' with 1 pod. The labels for this deployment are app=d1 and desarrollo. A command prompt window at the bottom shows the following commands:

```
PS C:\Users\g10n-> oc get deploy d1 -o wide --show-labels
oc label deploy d1 "responsable=pepe"
```

```
oc get deploy d1 -o wide --show-labels
oc label deploy d1 "responsable=pepe"
```

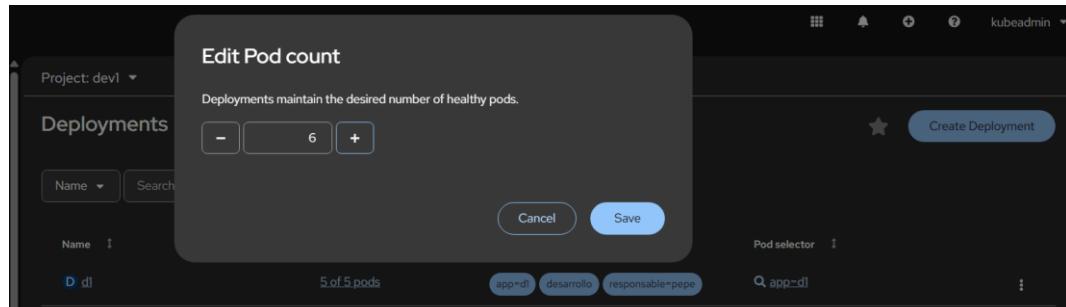
```
PS C:\Users\g10n-> oc get deploy d1 -o wide --show-labels
error: unknown flag: --show-labels
See 'oc get --help' for usage.
PS C:\Users\g10n-> oc get deploy d1 -o wide --show-labels
NAME    READY  UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES   SELECTOR   LABELS
d1     1/1     1           1          3h41m   httpd       httpd   app=d1   app=d1,desarrollo=
PS C:\Users\g10n-> oc label deploy d1 "responsable=pepe"
deployment.apps/d1 labeled
PS C:\Users\g10n-> oc get deploy d1 -o wide --show-labels
NAME    READY  UP-TO-DATE   AVAILABLE   AGE    CONTAINERS   IMAGES   SELECTOR   LABELS
d1     1/1     1           1          3h42m   httpd       httpd   app=d1   app=d1,desarrollo=,responsable=pepe
```

Escalar deployment

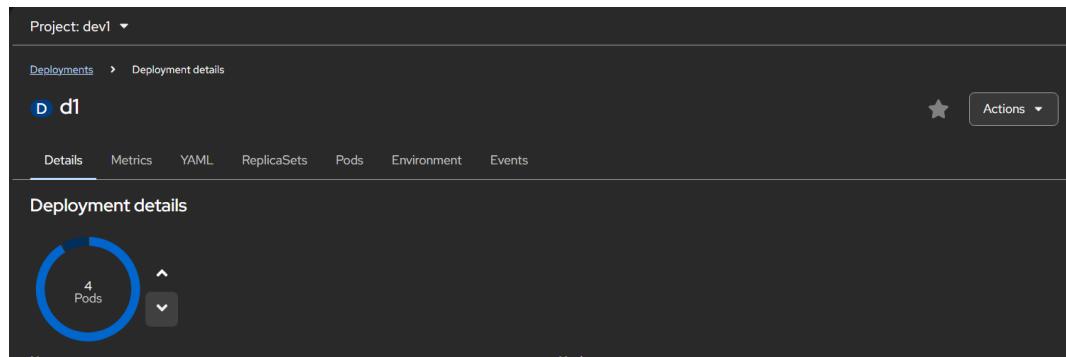
```
oc scale deploy d1 --replicas=5
```

```
PS C:\Users\g10n_> oc scale deploy d1 --replicas=5
deployment.apps/d1 scaled
PS C:\Users\g10n_> oc get pods
NAME           READY   STATUS    RESTARTS   AGE
d1-7956fbf4db-5fln5  1/1     Running   0          12s
d1-7956fbf4db-74kp5  1/1     Running   0          3h44m
d1-7956fbf4db-hzmv5  1/1     Running   0          12s
d1-7956fbf4db-l5thg  1/1     Running   0          12s
d1-7956fbf4db-l6wtb  1/1     Running   0          12s
nginx1          1/1     Running   0          5h10m
```

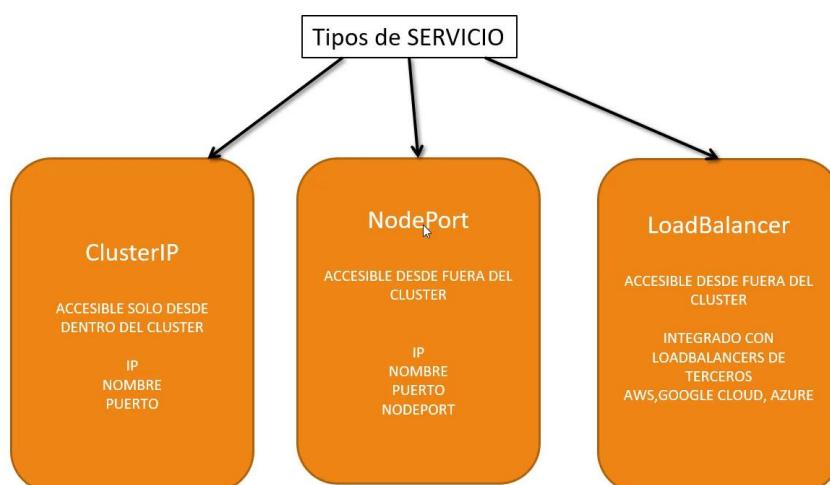
Aumentar desde la web



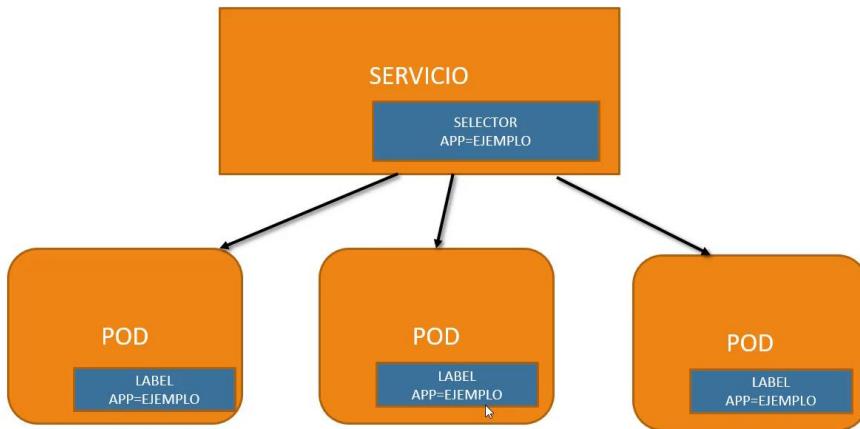
Aumentar o disminuir desde dentro del deployment, en el circulo



Servicios



El servicio apunta a los labels de los pods asociados



```
oc expose --name=apache1-svc deploy apache1 --type=NodePort
```

A screenshot of the OpenShift web console interface. On the left, a sidebar menu shows various project sections like Home, Favorites, Operators, Helm, Workloads, Networking, Services (which is selected), Routes, Ingresses, NetworkPolicies, UserDefinedNetworks, Storage, and Builds. The main area is titled "Create Service" with the sub-instruction "Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor." Below this is a code editor window containing the following YAML configuration:

```
apiVersion: v1
kind: Service
metadata:
  name: example
  namespace: dev1
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

At the bottom of the code editor are "Create" and "Cancel" buttons, and a "Download" button on the right.

```
oc create deploy nginx-dep --image=nginx -n=dev1
oc get pods -o wide --show-labels
#label=nginx-dep
```

En la web crear el siguiente servicio en yaml:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-svc
  namespace: dev1
spec:
  selector:
    app: nginx-dep
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 31800
  type: NodePort
```

Estaba apareciendo este error en el pod

```
0/1 nodes are available: 1 node(s) had untolerated taint {node.kubernetes.io/disk-pressure: }. preemption: 0/1 nodes are available: 1 Preemption is not helpful for scheduling.
```

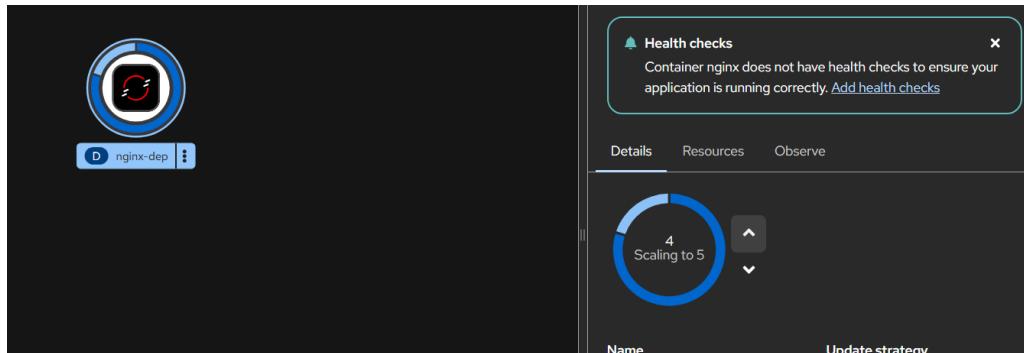
Ademas

```
PS C:\Users\g10n_\OneDrive\Escritorio\Curso Platzi\DevOps\11. Kubernetes\ku
● kubernetes> k get pods -n dev1
  NAME        READY   STATUS      RESTARTS   AGE
  nginx-dep-5879cc9-4nmdq   0/1     Completed   0          2m57s
  nginx-dep-5879cc9-4tr6f   0/1     ContainerStatusUnknown   0          2m47s
  nginx-dep-5879cc9-5jmwz   0/1     ContainerStatusUnknown   0          2m49s
  nginx-dep-5879cc9-5zbsl   0/1     ContainerStatusUnknown   0          2m49s
  nginx-dep-5879cc9-6dbhh   0/1     ContainerStatusUnknown   0          2m48s
● nginx-dep-5879cc9-74xkw   0/1     ContainerStatusUnknown   0          2m47s
  nginx-dep-5879cc9-7lsgn   0/1     ContainerStatusUnknown   0          2m49s
  nginx-dep-5879cc9-c26t4   0/1     ContainerStatusUnknown   0          2m49s
```

Se soluciono este error eliminando los pods que ocupaban espacio innecesario en el cluster

```
oc delete pods -n dev1 --field-selector=status.phase!=Running
```

Creando mas pods desde la web



```
PS C:\Users\g10n_> oc get pods -n dev1
  NAME        READY   STATUS      RESTARTS   AGE
  nginx-dep-5879cc9-4qkdd   1/1     Running   0          30s
  nginx-dep-5879cc9-f72rv   1/1     Running   0          31s
  nginx-dep-5879cc9-f9jtv   1/1     Running   0          31s
  nginx-dep-5879cc9-wjwpw   1/1     Running   0          29s
  nginx-dep-5879cc9-z6p2t   1/1     Running   0          14m
```

Podemos verificar que el servicio habilita correctamente los endpoints para cada pod

```
PS C:\Users\g10n_> oc describe svc nginx-svc -n dev1
Name:           nginx-svc
Namespace:      dev1
Labels:          <none>
Annotations:    <none>
Selector:       app=nginx-dep
Type:           NodePort
IP Family Policy: SingleStack
IPs:            IPv4
IP:             10.217.5.38
IPs:            10.217.5.38
Port:           <unset>  80/TCP
TargetPort:     80/TCP
NodePort:       <unset>  31800/TCP
Endpoints:      10.217.0.144:80,10.217.0.149:80,10.217.0.150:80 + 2 more...
Session Affinity: None
External Traffic Policy: Cluster
Internal Traffic Policy: Cluster
Events:          <none>
PS C:\Users\g10n_> oc get pods -o wide -n dev1
  NAME        READY   STATUS      RESTARTS   AGE   IP          NODE   NOMINATED NODE   READINESS GATES
  nginx-dep-5879cc9-4qkdd   1/1   Running   0      2m14s  10.217.0.151  crc   <none>        <none>
  nginx-dep-5879cc9-f72rv   1/1   Running   0      2m15s  10.217.0.150  crc   <none>        <none>
  nginx-dep-5879cc9-f9jtv   1/1   Running   0      2m15s  10.217.0.149  crc   <none>        <none>
  nginx-dep-5879cc9-wjwpw   1/1   Running   0      2m13s  10.217.0.152  crc   <none>        <none>
  nginx-dep-5879cc9-z6p2t   1/1   Running   0      15m    10.217.0.144  crc   <none>        <none>
```

Aumentar los recursos en CRC local *

Verificar los recursos actuales

```
crc config view
```

```
(TraeAI-3) C:\Users\g10n_\OneDrive\Escritorio\Curso Platzi\DevOps\11. Kubernetes\kubernetes [1:1] $ kubectl get node crc -o jsonpath='{.status.capacity}' | ConvertFrom-Json

cpu          : 4
ephemeral-storage : 31914988Ki
hugepages-1Gi   : 0
hugepages-2Mi    : 0
memory         : 10695244Ki
pods           : 250
```

Capacidad de CRC actual:

- CPU : 4 cores
- Memoria : ~10.4GB (10,695,244 Ki)
- Almacenamiento : ~31GB (31,914,988 Ki)
- Pods máximos : 250

Script para aumentar capacidades

```
# 1. Detener CRC
crc stop

# 2. Configurar más recursos
crc config set memory 16384    # 16GB RAM
crc config set cpus 6        # 6 CPU cores
crc config set disk-size 50    # 50GB disco

# 3. Reiniciar CRC
crc start
```

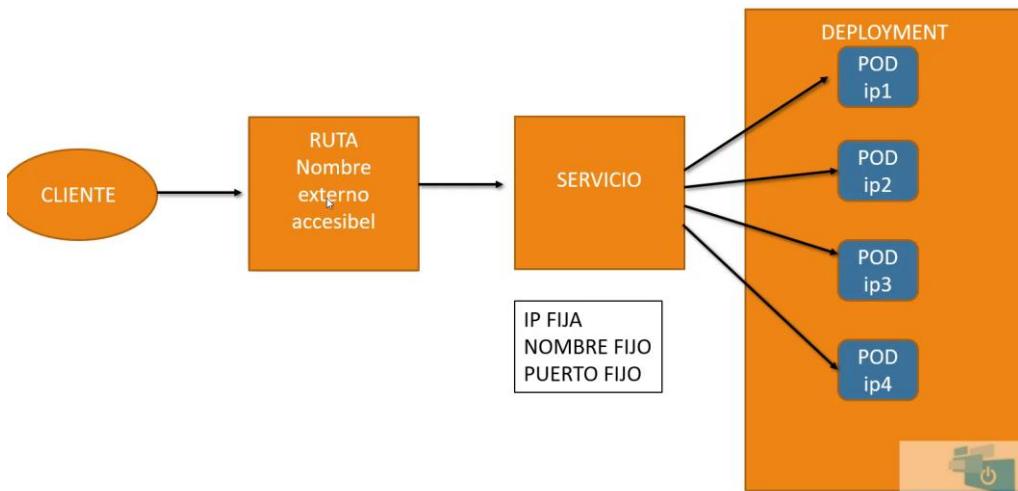
Estado actual

```
○ PS C:\Users\g10n_\OneDrive\Escritorio\Curso Platzi\DevOps\11. Kubernetes\kub
● ernetes> crc config view
- consent-telemetry          : no
- cpus                         : 6
- disk-size                    : 50
- memory                       : 16384
○ PS C:\Users\g10n_\OneDrive\Escritorio\Curso Platzi\DevOps\11. Kubernetes\kub
ernetes> []
```

Routes (propio de Openshift)

■ Rutas

- Una ruta es un objeto que permite exponer una aplicación web al tráfico externo en un clúster de OpenShift.
- Crea una URL que redirige las solicitudes de los clientes al servicio subyacente y por tanto al deploy o al deploymentConfig de la aplicación.
- Es por tanto, un punto de entrada para nuestras aplicaciones
- Son muy parecidas a los Ingress controller de Kubernetes.
- De hecho, podemos seguir usando Ingress en un cluster de OpenShift



Crear ruta usando la linea de comandos

```
oc expose svc web1-svc -n dev1
```

```
PS C:\Users\g10n_> oc get svc -n dev1
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
nginx-svc  NodePort  10.217.5.38    <none>          80:31800/TCP  47m
web1-svc   NodePort  10.217.5.129   <none>          80:30002/TCP  27m
PS C:\Users\g10n_> oc expose svc web1-svc -n dev1
route.route.openshift.io/web1-svc exposed
PS C:\Users\g10n_> oc get route -n dev1
NAME      HOST/PORT      PATH      SERVICES      PORT      TERMINATION      WILDCARD
web1-svc  web1-svc-dev1.apps-crc.testing      web1-svc      80
PS C:\Users\g10n_> |
```

```
web1-svc-dev1.apps-crc.testing
```

Probarlo en la web como (sin considerar la S en http)

```
http://web1-svc-dev1.apps-crc.testing/
```

Tambien podemos verlo desde el admin de Openshift

Name	Status	Location	Service
RT web1-svc	Accepted	http://web1-svc-dev.apps-crc.testing	web1-svc

Crear ruta desde el formulario

Project: devl

Create Route

Routing is a way to make your application publicly visible

Configure via: Form view (radio button selected) YAML view

Name *

route-horizontal-cobra

A unique name for the Route within the project

Hostname

Path

Service *

Select a Service

Service *

S nginx-svc

Service to route to.

Service weight

100

A number between 0 and 255 that depicts relative weight compared with other targets.

Add alternate Service

Target port *

80 → 80 (TCP)

Target port for traffic

Ruta creada

Project: devl

Routes > Route details

RT apache1-ruta

Details Metrics YAML Actions

Route details

Name apache1-ruta

Namespace NS devl

Labels No labels

Location http://apache1-ruta-dev.apps-crc.testing

Status Accepted

Host apache1-ruta-dev.apps-crc.testing

Crear ruta con dominio personalizado (solo de prueba)

Project: dev1 ▾

Create Route
Routing is a way to make your application publicly visible

Configure via: Form view YAML view

Name *

A unique name for the Route within the project

Hostname

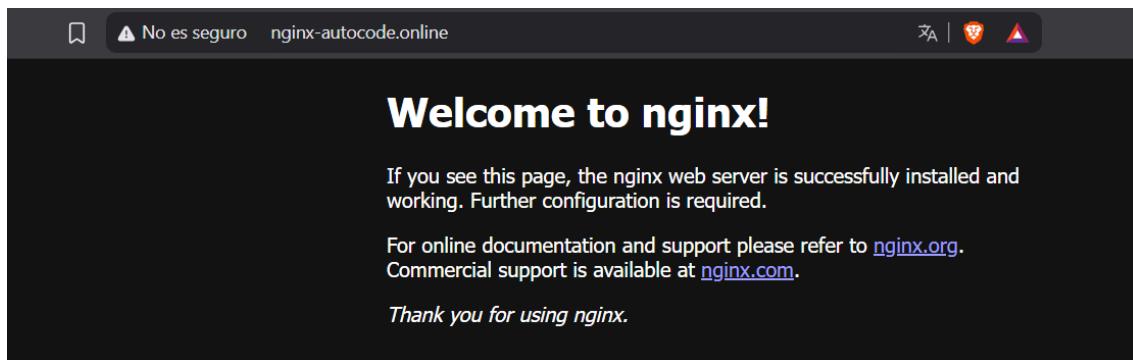
Public hostname for the Route. If not specified, a hostname is generated.

Agregar la ruta en el archivo hosts de Windows

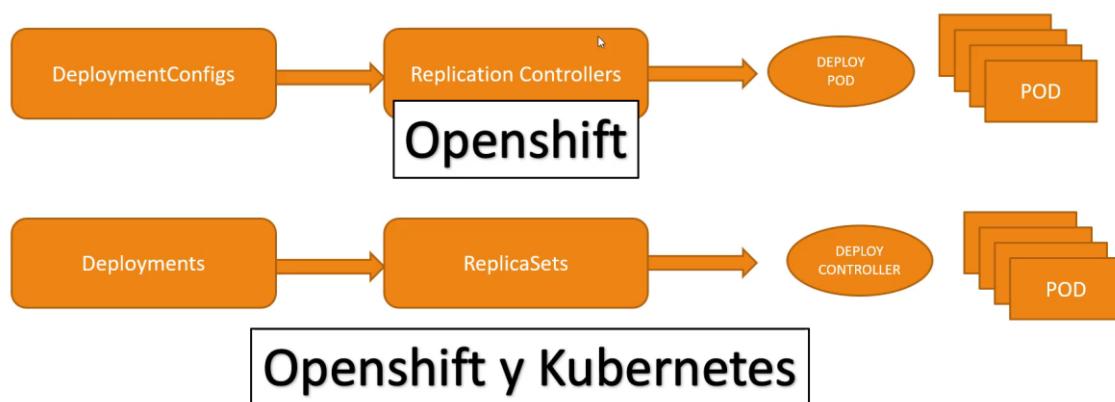
C:\Windows\System32\drivers\etc\hosts

127.0.0.1 nginx-autocode.online

Probar nuevamente la URL personalizada



DeploymentConfigs (Deprecados)



-
- Características de un DeploymentConfig
 - Un DeploymentConfig, es en realidad una plantilla para lanzar aplicaciones.
 - Dispone de Triggers que disparan despliegues automáticos ante determinados eventos.
 - Existen estrategias configurables para determinar la transición desde la versión anterior a la actual.
 - Hay una serie de hooks (lifecycle hooks) que permiten configurar un comportamiento en diferentes momentos del ciclo de vida del despliegue.
 - Versionado de la aplicación, para poder realizar rollbacks tanto de forma manual como automática.
 - Se soporta escalado manual y automático de las réplicas.



- Diferencias con los Deployment de Kubernetes
 - Los DeploymentConfigs se basan en la consistencia mientras que los Deployments prefieren la disponibilidad
 - Automatic rollbacks: los Deployments no tienen rolling back automático al último ReplicaSet desplegado en caso de fallo.
 - Lifecycle hooks: Los Deployments ^{No}soportan lifecycle hooks.
 - Custom strategies: Los Deployments no disponen de estrategias personalizadas de despliegue.



- Diferencias con los Deployment de Kubernetes
- ROLLOVER
 - Los Deployment pueden tener tantos ReplicaSets activos como sea necesarios, de forma que el deployment controller gestionara la bajada de los antiguos y la subida de los nuevos.
 - Los DeploymentConfigs solo pueden tener un deployer pod funcionando. Por tanto, solo 2 ReplicationControllers pueden estar activos al mismo tiempo.

-
- Diferencias con los Deployment de Kubernetes
 - ESCALADO PROPORCIONAL
 - Dado que el Deployment controller es el que gestiona los tamaños de los nuevos y de los viejos ReplicaSets se pueden escalar los rollouts que en ese momento estén en marcha..
 - DeploymentConfigs no pueden ser escalados mientras estamos en un rollout porque el DeploymentConfig controller no es capaz de gestionarlo.

Comandos

```
create deploymentconfig apache-dc --image=httpd  
oc get rc # Replication Controller  
oc get pod # Crea un pod adicional
```

```
openshift-->oc create deploymentconfig apache-dc --image=httpd  
deploymentconfig.apps.openshift.io/apache-dc created  
openshift-->oc get dc  
NAME      REVISION  DESIRED  CURRENT  TRIGGERED BY  
apache-dc  1          1         1        config  
openshift-->oc get rc  
NAME      DESIRED  CURRENT  READY    AGE  
apache-dc-1 1          1         1        41s  
openshift-->oc get pod  
NAME           READY  STATUS    RESTARTS  AGE  
apache-dc-1-deploy  0/1   Completed  0         73s  
apache-dc-1-zkfvw  1/1   Running   0         66s  
apache1-89d978569-fkss2 1/1   Running   2         21h  
apache1-89d978569-vft17 1/1   Running   2         21h  
d1-67d4b54b64-gdnmb  1/1   Running   2         23h  
d2-9f46d8bf9-vc767  1/1   Running   2         22h  
nginxx-dep-7ccf5fddd7-1bn7p 1/1   Running   2         17h  
nginxx-dep-7ccf5fddd7-xkrqx 1/1   Running   2         17h  
web1-cf7cc764f-1g1pw  1/1   Running   2         16h  
web1-cf7cc764f-zwgdt  1/1   Running   2         16h  
openshift-->
```

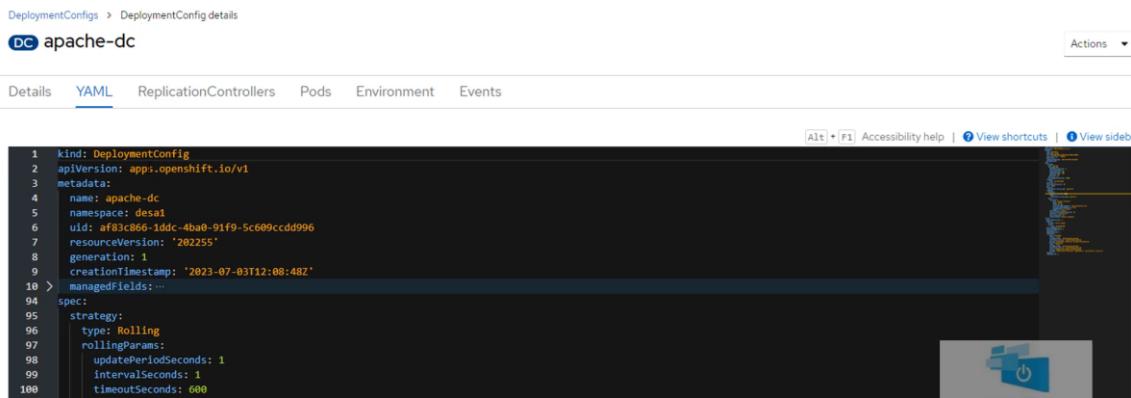
DeploymentConfigs > DeploymentConfig details

DC apache-dc

Actions ▾

Details **YAML** ReplicationControllers Pods Environment Events

Alt + F1 Accessibility help | View shortcuts | View sidebar



```
1 kind: DeploymentConfig  
2 apiVersion: apps.openshift.io/v1  
3 metadata:  
4   name: apache-dc  
5   namespace: default  
6   uid: af83cb66-1ddc-4ba0-9ff9-5c609ccdd996  
7   resourceVersion: '20225'  
8   generation: 1  
9   creationTimestamp: '2023-07-03T12:08:48Z'  
10  managedFields:...  
94  spec:  
95    strategy:  
96      type: Rolling  
97      rollingParams:  
98        updatePeriodSeconds: 1  
99        intervalSeconds: 1  
100       timeoutSeconds: 600
```

Desde la web

Tiene una opción más de estrategia de deployment, que es la de custom

Configure via: Form view YAML view

Note: Some fields may not be represented in this form view. Please select "YAML view" for full control.

Name *

Deployment strategy

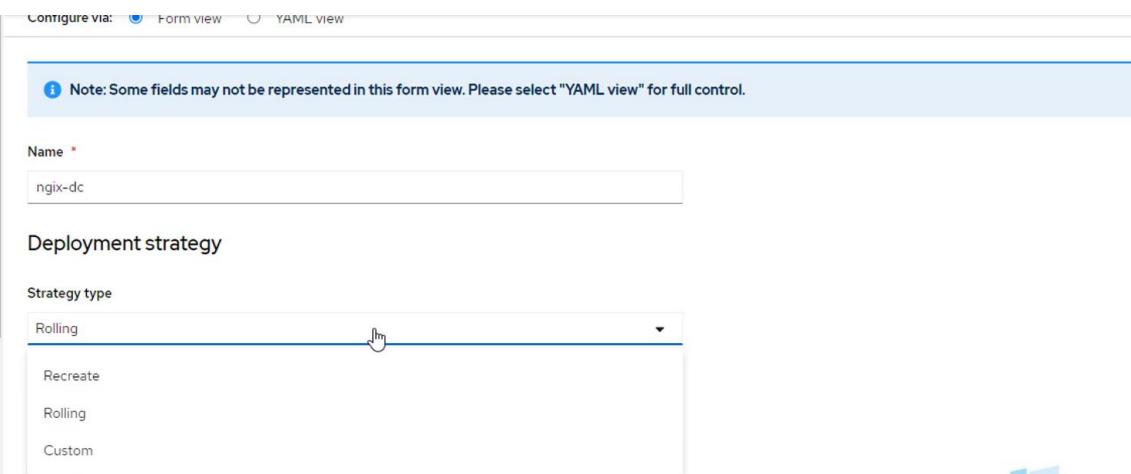
Strategy type

Rolling

Recreate

Rolling

Custom



Crear un servicio y ruta al DC

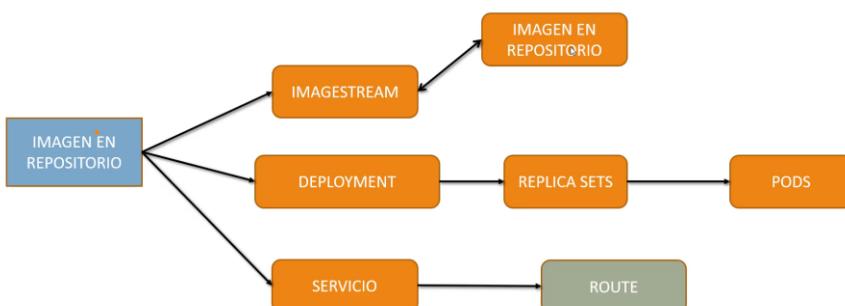
```
openshift-->oc get dc
NAME      REVISION  DESIRED  CURRENT  TRIGGERED BY
apache-dc  1         1         1         config
nginx-dc   1         2         2         config
openshift-->oc expose --name=apache-dc-svc deploymentconfig apache-dc --port=80
service/apache-dc-svc exposed
openshift-->oc get svc
NAME          TYPE    CLUSTER-IP      EXTERNAL-IP      PORT(S)        AGE
apache-dc-svc ClusterIP  10.217.5.32   <none>        80/TCP        7s
apache1-svc   NodePort  10.217.5.53   <none>        80:31767/TCP  26h
nginx-svc    NodePort  10.217.4.253  <none>        80:31800/TCP  25h
web1-svc     NodePort  10.217.5.105  <none>        80:30002/TCP  24h
openshift-->oc expose svc apache-dc-svc --name ruta-apache-svc
route.route.openshift.io/ruta-apache-svc exposed
openshift-->oc get route
NAME          HOST/PORT          PATH  SERVICES      PORT  TERMINATION  WILDCARD
apache-ruta   www.apasoft-prueba.com
ruta-apache-svc ruta-apache-svc-desal.apps-crc.testing
web1-svc      web1.svc-desal.apps-crc.testing
openshift-->
```

Los pods que tienen rutas configuradas tienen un icono de “Open Url”

Despliegue de aplicaciones

- ❑ Las aplicaciones se pueden desplegar en OpenShift de varias maneras posibles, tanto desde la consola web como del cliente OC.
- ❑ Los principales métodos para implementar una aplicación son:
 - ❑ Desplegar una aplicación desde una imagen existente.
 - ❑ Construir y desplegar una aplicación desde código fuente contenido en un repositorio GIT y usando Source-to-Image builder (S2I)
 - ❑ Construir y desplegar una aplicación desde código fuente contenido en un repositorio GIT y usando un Dockerfile.
 - ❑ Desde plantillas

❑ OBJETOS GENERADOS EN UN DEPLOY DE IMAGEN



Crear varios recursos en el cluster con Openshift

```
oc new-app apasoft/blog
```

```
PS C:\Users\g10n_> oc new-app apasoft/blog
--> Found container image 5ada8e2 (5 years old) from Docker Hub for "apasoft/blog"

  Python 3.5
  -----
  Python 3.5 available as container is a base platform for building and running various Python 3.5 applications and frameworks. Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

  Tags: builder, python, python35, python-35, rh-python35
  * An image stream tag will be created as "blog:latest" that will track this image

--> Creating resources ...
  imagestream.image.openshift.io "blog" created
  deployment.apps "blog" created
  service "blog" created
--> Success
  Application is not exposed. You can expose services to the outside world by executing one or more of the commands below:
    'oc expose service/blog'
  Run 'oc status' to view your app.
PS C:\Users\g10n_> |
```

```
oc expose service/blog
```

Consultar ImageStream

```
oc get is
```

```
route.route.openshift.io/blog-exposed
PS C:\Users\g10n_> oc get is
NAME      IMAGE REPOSITORY
blog      default-route-openshift-image-registry.apps-crc.testing/dev1/blog   TAGS      UPDATED
PS C:\Users\g10n_> |
```

Describir objeto

```
oc describe is blog
```

```
PS C:\Users\g10n_> oc describe is blog
Name:          blog
Namespace:     dev1
Created:       2 minutes ago
Labels:        app=blog
               app.kubernetes.io/component=blog
               app.kubernetes.io/instance=blog
Annotations:   openshift.io/generated-by=OpenShiftNewApp
               openshift.io/image.dockerRepositoryCheck=2025-10-14T18:09:03Z
Image Repository: default-route-openshift-image-registry.apps-crc.testing/dev1/blog
Image Lookup:   local=false
Unique Images:  1
Tags:          1
```

Ver todas las imágenes del cluster

```
oc get images
```

Filtrar imagen

```
oc get images | Select-String "blog" #windows
#oc get images | grep blog # Linux
```

```
PS C:\Users\g10n_> oc get images | Select-String "blog" #windows
sha256:9c8d3cedcd722ae6a9508ca532102501dfb796600c53a61d5178cdfaf22e02ed
apasoft/blog@sha256:9c8d3cedcd722ae6a9508ca532102501dfb796600c53a61d5178cdfaf22e02ed
```

```

PS C:\Users\g10n_> oc get deploy
NAME      READY  UP-TO-DATE  AVAILABLE  AGE
blog      1/1    1           1          7m46s
nginx-dep 1/1    1           1          21h
web1      2/2    2           2          21h
PS C:\Users\g10n_> oc get svc
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
blog      ClusterIP  10.217.4.251  <none>        8080/TCP    7m56s
nginx-svc NodePort   10.217.5.38   <none>        80:31800/TCP  21h
web1-svc  NodePort   10.217.5.129  <none>        80:30002/TCP  21h
PS C:\Users\g10n_> oc get route
NAME      HOST/PORT      PATH      SERVICES      PORT      TERMINATION      WILDCARD
blog      blog-dev1.apps-crc.testing      blog      8080-tcp      None
nginx-ruta  nginx-autocode.online      nginx-svc      80      None
web1-svc  web1-svc-dev1.apps-crc.testing      web1-svc      80      None
PS C:\Users\g10n_> oc get pod
NAME      READY  STATUS  RESTARTS  AGE
blog-9d54ff9c7-ssz7m  1/1    Running  0          8m22s
nginx-dep-5879cc9-z6p2t 1/1    Running  2          21h
web1-fd4f6ff7-fddzc  1/1    Running  2          21h
web1-fd4f6ff7-hw4vb  1/1    Running  2          21h
PS C:\Users\g10n_>

```

March 14, 2017, 3:26 a.m.

What is OpenShift Origin?

Origin is the upstream community project that powers OpenShift. Built around a core of Docker container packaging and Kubernetes container cluster management, Origin is also augmented by application lifecycle management functionality and DevOps tooling. Origin provides a complete open source container application platform.

En la web Topology

Nueva aplicación con DeploymentConfig

```

oc new-app --name=nginx10 --as-deployment-config=true --image=nginx
oc get dc
oc get rc
oc get pod

```

Desplegar la imagen nextcloud

```

oc new-app --name=mi-nube nextcloud

```

```

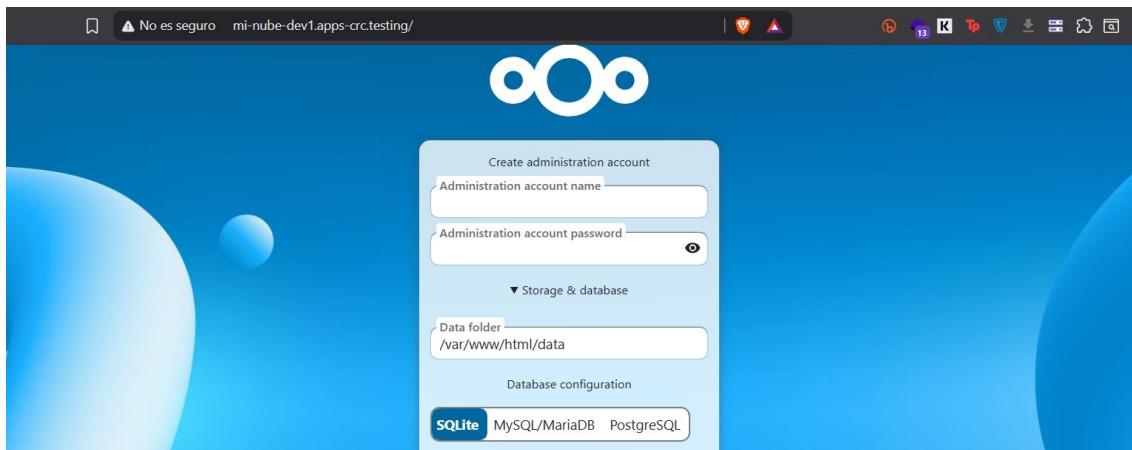
PS C:\Users\g10n_> oc new-app --name=mi-nube nextcloud
--> Found container image b97ea40 (2 weeks old) from Docker Hub for "nextcloud"
    * An image stream tag will be created as "mi-nube:latest" that will track this image

--> Creating resources ...
imagestream.image.openshift.io "mi-nube" created
deployment.apps "mi-nube" created
service "mi-nube" created
--> Success
Application is not exposed. You can expose services to the outside world by executing one or more of the commands below:
  'oc expose service/mi-nube'
Run 'oc status' to view your app.
PS C:\Users\g10n_> oc expose service/mi-nube
route.route.openshift.io/mi-nube exposed
PS C:\Users\g10n_> oc get svc mi-nube
NAME           IMAGE REPOSITORY
mi-nube        default-route-openshift-image-registry.apps-crc.testing/dev1/mi-nube   latest   30 seconds ago
PS C:\Users\g10n_> oc get svc mi-nube
NAME           TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
mi-nube        ClusterIP 10.217.4.175  <none>          80/TCP       62s
PS C:\Users\g10n_> oc get route mi-nube
NAME          HOST/PORT      PATH      SERVICES      PORT      TERMINATION      WILDCARD
mi-nube       mi-nube-dev1.apps-crc.testing   mi-nube     80-tcp      None
PS C:\Users\g10n_> oc get all -l app=mi-nube
Warning: apps.openshift.io/v1 DeploymentConfig is deprecated in v4.14+, unavailable in v4.18.00+
NAME          TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
service/mi-nube ClusterIP 10.217.4.175  <none>          80/TCP       93s

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mi-nube 1/1      1           1           93s

NAME           IMAGE REPOSITORY
imagestream.image.openshift.io/mi-nube  default-route-openshift-image-registry.apps-crc.testing/dev1/mi-nube   latest   About a minute ago
NAME          HOST/PORT      PATH      SERVICES      PORT      TERMINATION      WILDCARD
route.route.openshift.io/mi-nube     mi-nube-dev1.apps-crc.testing   mi-nube     80-tcp      None
PS C:\Users\g10n_>

```



Comprobar objetos generados sin lanzar aplicación

```
oc new-app --name=ejemplo --image=tomcat -o yaml > tomcat.yaml
```

Se puede entrar al yaml para modificarlo

```

g10n92@Giancarlo:~$ oc new-app --name=ejemplo --image=tomcat -o yaml > tomcat.yaml
g10n92@Giancarlo:~$ oc get deploy
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
mi-nube  1/1      1           1           3d22h
g10n92@Giancarlo:~$ cat tomcat.yaml
apiVersion: v1
items:
- apiVersion: image.openshift.io/v1
  kind: ImageStream
  metadata:
    annotations:
      openshift.io/generated-by: OpenShiftNewApp
    creationTimestamp: null
  labels:
    app: ejemplo

```

Aplicar cambios

```
oc apply -f tomcat.yaml
```

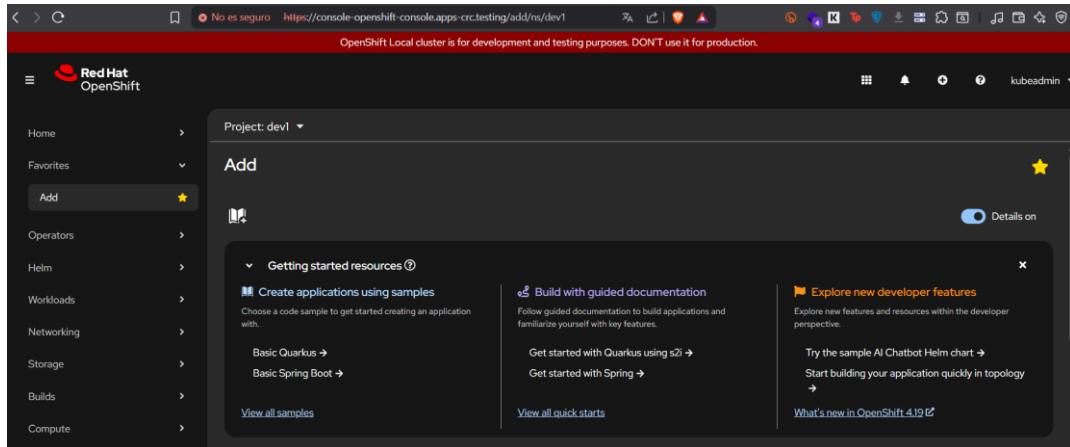
```

metadata: {}
g10n92@Giancarlo:~$ oc apply -f tomcat.yaml
imagestream.image.openshift.io/ejemplo created
deployment.apps/ejemplo created
service/ejemplo created

```

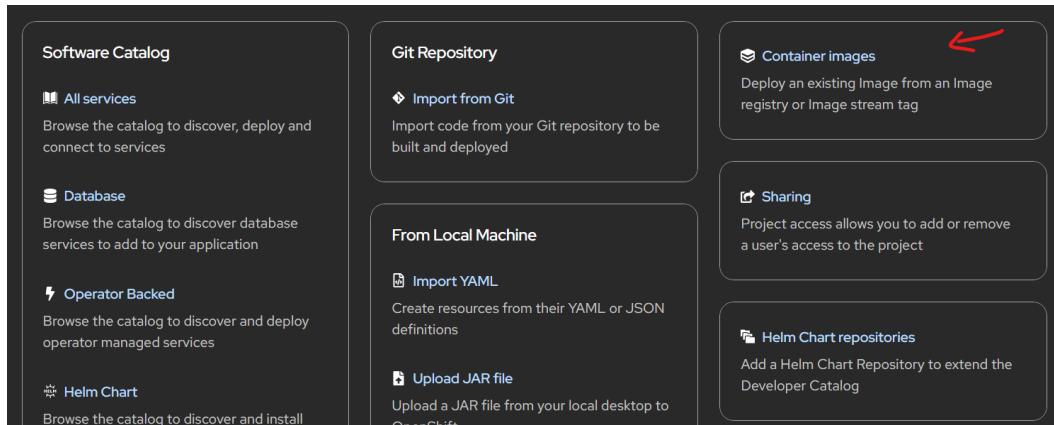
Desplegar proyectos desde la web

Entrar a la web desde la URL: add/ns/dev1[project], y agregarlo como favorito, porque no se encuentra facilmente



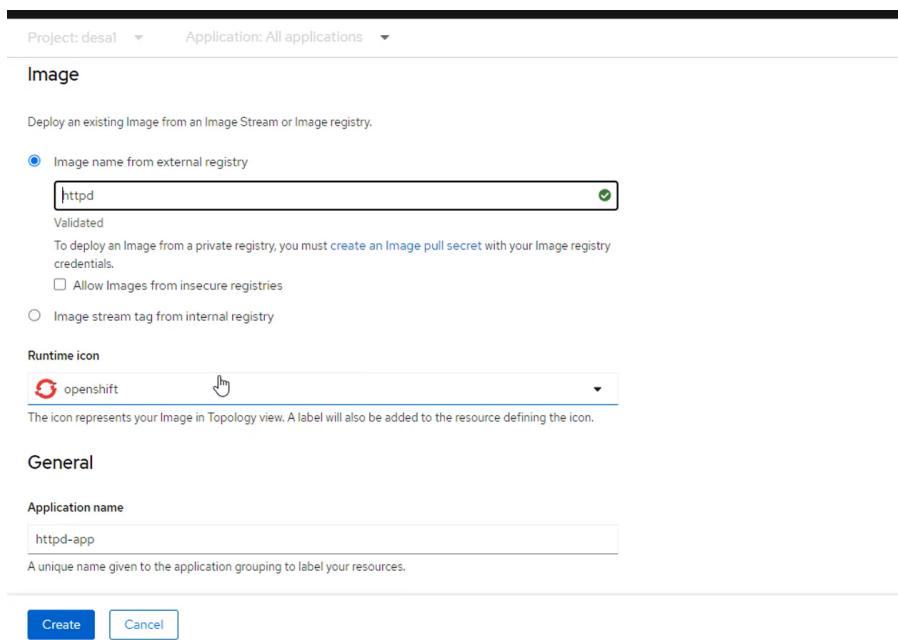
The screenshot shows the Red Hat OpenShift console interface. The top navigation bar has 'No es seguro' and the URL 'https://console.openshift-console.apps-crc.testing/add/ns/dev1'. Below the navigation is a banner stating 'OpenShift Local cluster is for development and testing purposes. DON'T use it for production.' The main area is titled 'Project: dev1' and has a 'Add' button. A sidebar on the left lists categories like Home, Favorites, Operators, Helm, Workloads, Networking, Storage, Builds, and Compute. The 'Add' section contains 'Getting started resources' with links to 'Create applications using samples' (Basic Quarkus, Basic Spring Boot), 'Build with guided documentation' (Get started with Quarkus using s2i, Get started with Spring), and 'Explore new developer features' (Try the sample AI Chatbot Helm chart, Start building your application quickly in topology). A yellow star icon is placed next to the 'Add' button.

Luego vamos a elegir “Container images”



The screenshot shows the 'Developer Catalog' interface. It has a sidebar with 'Software Catalog' sections for 'All services', 'Database', 'Operator Backed', and 'Helm Chart'. The main area has sections for 'Git Repository' (Import from Git), 'From Local Machine' (Import YAML, Upload JAR file), 'Sharing' (Project access), and 'Helm Chart repositories'. A red arrow points to the 'Container images' section, which is described as 'Deploy an existing Image from an Image registry or Image stream tag'.

Poner el nombre de la imagen, por defecto busca en docker hub, se puede elegir icono

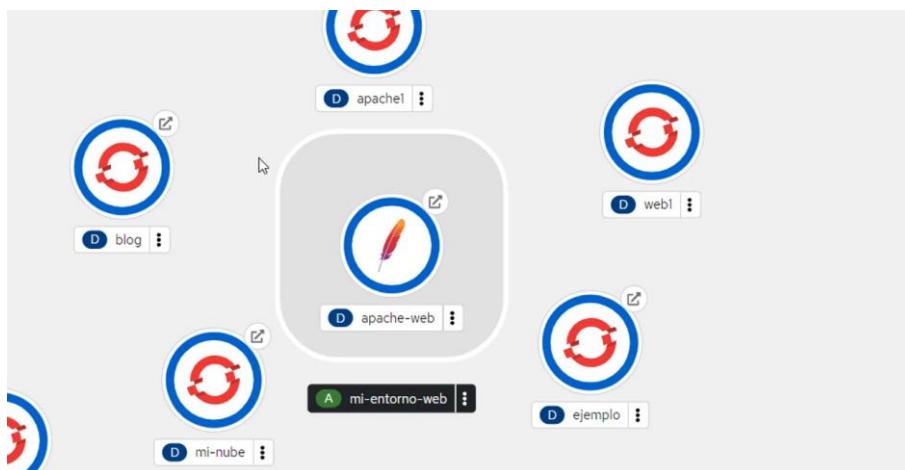


The screenshot shows the 'Image' configuration screen. At the top, there are dropdowns for 'Project: desal' and 'Application: All applications'. The main area has tabs for 'Image', 'General', and 'Advanced'. The 'Image' tab is active, showing a field for 'Image' with 'httpd' entered, a 'Runtime icon' dropdown with 'openshift' selected, and a note about creating an 'Image pull secret'. The 'General' tab shows an 'Application name' field with 'httpd-app' and a note about labeling resources. At the bottom are 'Create' and 'Cancel' buttons.

Se puede elegir otras configuraciones

The screenshot shows the configuration page for creating a new application. At the top, there's a field for the application name: 'mi-entorno-web'. Below it, a note says: 'A unique name given to the application grouping to label your resources.' Under 'Name *', the component name is 'apache-web', with the note: 'A unique name given to the component that will be used to name associated resources.' A 'Resource type' dropdown is set to 'Deployment', with 'DeploymentConfig' selected. A note below says: 'A DeploymentConfig defines the template for a Pod and manages deploying new Images or configuration changes.' There's also a 'Target port for traffic' field set to '80'. A checked checkbox 'Create a route' is followed by the note: 'Exposes your component at a public URL.' A link 'Show advanced Routing options' is present. At the bottom are 'Create' and 'Cancel' buttons.

Para la versión web podemos ver el nombre de la aplicación como esta arriba, dentro de este puede existir varios deployments



Crear desde una imagen interna

This screenshot shows the 'Create from Internal Image' configuration page. It includes fields for 'Project' (set to 'desal'), 'Image Stream' (dropdown 'Select Image Stream'), 'Tag' (dropdown 'Select tag'), 'Runtime icon' (set to 'openshift'), and an 'Application' field containing 'mi-entorno-web'. A note below says: 'Select an Application to group this component'. A 'Name *' field is also present. On the right, a sidebar lists various application names: 'blog', 'ejemplo', 'jboss', 'mariadb', 'mi-nube', 'php1' (with a cursor icon), and 'wildfly'. A note at the bottom right says: 'A unique name given to the component that will be used to name associated resources.'

El proyecto openshift ya viene con varias imagenes para reutilizar

Image stream tag from internal registry

Project * openshift / Image Stream * Tag *

Runtime icon openshift

The icon represents your Image in Topology

General

Application mi-entorno-web

Select an Application to group this component

Name *

Create Cancel

jboss-webserver57-openjdk11-tomcat9-openshift-ubi8
jboss-webserver57-openjdk8-tomcat9-openshift-ubi8
jenkins
jenkins-agent-base
mariadb
must-gather
mysql
network-tools
nginx
nodejs
oauth-proxy

Se puede consultar desde la consola tambien

```
oc get is -n openshift
```

```
openshift-->oc get is
NAME          IMAGE REPOSITORY
apache-web    default-route-openshift-image-registry.apps-crc.testing/desal/apache-web
blog          default-route-openshift-image-registry.apps-crc.testing/desal/blog
ejemplo       default-route-openshift-image-registry.apps-crc.testing/desal/ejemplo
jboss         default-route-openshift-image-registry.apps-crc.testing/desal/jboss
mariadb       default-route-openshift-image-registry.apps-crc.testing/desal/mariadb
mi-nube       default-route-openshift-image-registry.apps-crc.testing/desal/mi-nube
php1          default-route-openshift-image-registry.apps-crc.testing/desal/php1
wildfly       default-route-openshift-image-registry.apps-crc.testing/desal/wildfly
openshift-->oc get is -n openshift
```

Buscar uno en concreto

```
openshift-->oc get is -n openshift | grep mariadb
mariadb      default-route-openshift-image-registry.apps-crc.testing/openshift/mariadb
              10.3.10.3-e17.10.3-e18.10.5-e17 +3 more...
              7 weeks ago
openshift-->
```

Cada TAG apunta a una imagen concreta de MariaDB

Agrupacion de la aplicaciones

Topology

Observe

Search

Builds

Helm

Project

ConfigMaps

Secrets

correctly. Add health checks

Details Resources Observe

Pods

mariadb1-8755dd5d9-vbgpn Creating View logs

Services

mariadb Service port: 3306-tcp → Pod port: 3306

Routes

mariadb Location: https://mariadb-desal.apps-crc.testing:3306 www.apachefortraining.com

Comandos interesantes

Entrar a la consola de pod

```
oc rsh pod_name
```

Modificar en caliente el archivo de openshift (se despliega de nuevo el pod)

```
oc edit dc ejemplo
```

Esto se hace porque tiene un trigger configurado

```
terminationGracePeriodSeconds: 30
test: false
triggers:
- type: ConfigChange
status:
```

Ver variables de entorno de un recurso de openshift como DC, PODs, etc

```
oc set env pod/ejemplo-variables --list
oc set env dc/ejemplo-variables --list
```

```
[openshift@curso Variables]$ oc set env poc/ejemplo-variables-2-ljml9 --list
error: the server doesn't have a resource type "poc"
[openshift@curso Variables]$ oc set env pod/ejemplo-variables-2-ljml9 --list
# pods/ejemplo-variables-2-ljml9, container variables
NOMBRE=CURSO DE OPENSHIFT
PROPIETARIO=Apasoft Training
VERSION=v1
[openshift@curso Variables]$ oc set env dc/ejemplo-variables --list
# deploymentconfigs/ejemplo-variables, container variables
NOMBRE=CURSO DE OPENSHIFT
PROPIETARIO=Apasoft Training
VERSION=v1
[openshift@curso Variables]$
```

Agregar nuevas variables

```
oc set env pod/ejemplo-variables RESPONSABLE=pedro
oc set env pod/ejemplo-variables --list
```

Se vuelven a desplegar nuevos pods

```
[openshift@curso Variables]$ oc set env dc/ejemplo-variables RESPONSABLE=pedro
deploymentconfig.apps.openshift.io/ejemplo-variables updated
[openshift@curso Variables]$ oc set env dc/ejemplo-variables --list
# deploymentconfigs/ejemplo-variables, container variables
NOMBRE=CURSO DE OPENSHIFT
PROPIETARIO=Apasoft Training
VERSION=v1
RESPONSABLE=pedro
[openshift@curso Variables]$ oc get pod
NAME          READY   STATUS    RESTARTS   AGE
ejemplo-variables-1-deploy  0/1     Completed   0          13m
ejemplo-variables-2-9xm6d  1/1     Terminating 0          11m
ejemplo-variables-2-deploy 0/1     Completed   0          11m
ejemplo-variables-2-ljml9  1/1     Terminating 0          10m
ejemplo-variables-3-deploy 0/1     Completed   0          25s
ejemplo-variables-3-gf8kq  1/1     Running    0          20s
ejemplo-variables-3-hpwtw  1/1     Running    0          16s
[openshift@curso Variables]$
```

Sobre escribir variables

```
oc set env pod/ejemplo-variables --overwrite RESPONSABLE=rosa  
oc set env pod/ejemplo-variables --list
```

Recordar que esto lanza una nueva version

```
[openshift@curso Variables]$ oc set env dc/ejemplo-variables --list  
# deploymentconfigs/ejemplo-variables, container variables  
NOMBRE=CURSO DE OPENSHIFT  
PROPIETARIO=Apasoft Training  
VERSION=v1  
RESPONSABLE=Rosa  
[openshift@curso Variables]$ oc get dc  
NAME           REVISION  DESIRED  CURRENT  TRIGGERED BY  
ejemplo-variables  4       2        2        config
```

Quitar variables

```
oc set env pod/ejemplo-variables RESPONSABLE-  
oc set env pod/ejemplo-variables --list
```

```
[openshift@curso Variables]$ oc get dc  
NAME           REVISION  DESIRED  CURRENT  TRIGGERED BY  
ejemplo-variables  5       2        2        config  
[openshift@curso Variables]$
```

Tambien lo puedes ver con el comando edit

```
oc edit dc/ejemplo-variables
```

```
spec:  
  containers:  
    - env:  
      - name: NOMBRE  
        value: CURSO DE OPENSHIFT  
      - name: PROPIETARIO  
        value: Apasoft Training  
      - name: VERSION  
        value: v1  
      - name: RESPONSABLE  
        value: Rosa  
    image: gcr.io/google-samples/node-hello:1.0  
    imagePullPolicy: IfNotPresent  
  name: variables
```

