

03-Lab1 资源探索实验

场景

在这个实验室中，您将演示如何在Red Hat®OpenShift®容器平台上部署应用程序。您扮演一个开发人员，他希望从Git存储库部署一个Node.js示例应用程序，并查看为该应用程序创建的不同的OpenShift资源。

您将在OpenShift容器平台界面中采取**行动**，并**解释**：操作、产品功能、资源和概念。

目标

- 了解web控制台视角：开发人员和管理员
- 创建一个项目
- 从Git仓库中构建和部署应用程序
- 熟悉网络控制台的拓扑视图
- 将资源关联为应用分组的一部分
- 了解已部署资源的管理员视图

1. 提供实验室环境

请参考讲师的现场实验环境访问说明。

2. 演示OpenShift的身份验证

1. **行动**: 使用您的登录凭证登录到OpenShift容器平台web控制台。



Red Hat OpenShift Container Platform

Log in to your account

Username *

Password *

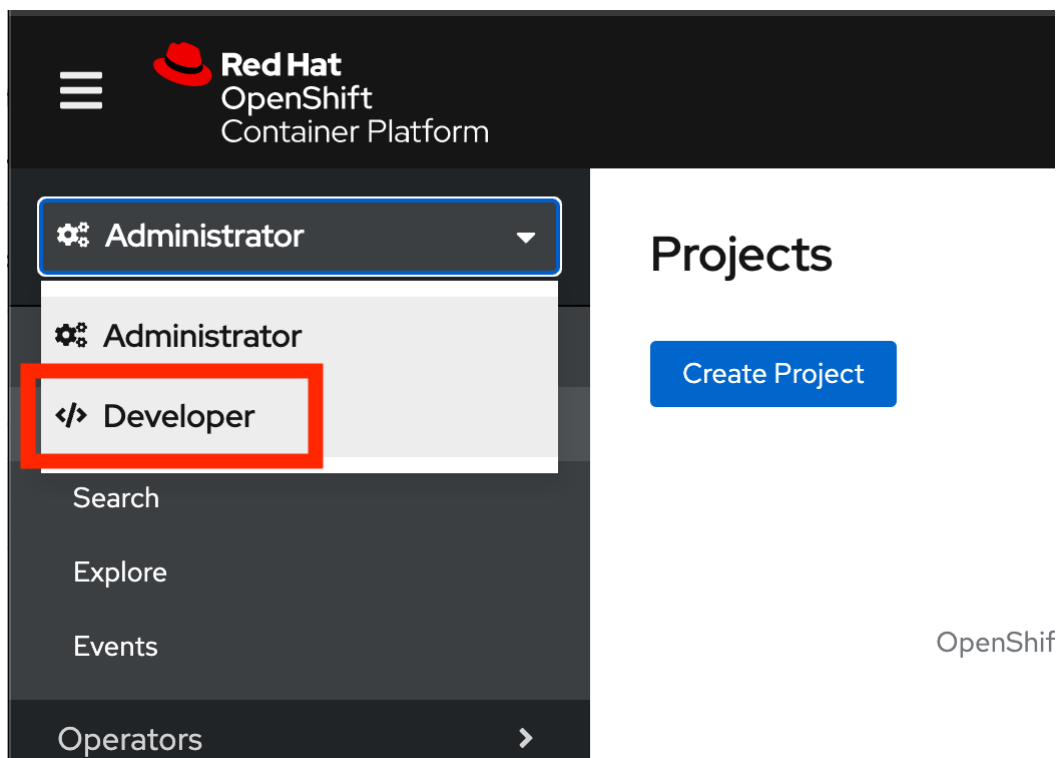
Log in

Welcome to Red Hat OpenShift Container Platform.

- **解释** 您被带到**项目**页面。OpenShift容器平台web控制台的默认视图是管理员视图。

3. 演示开发人员透视图

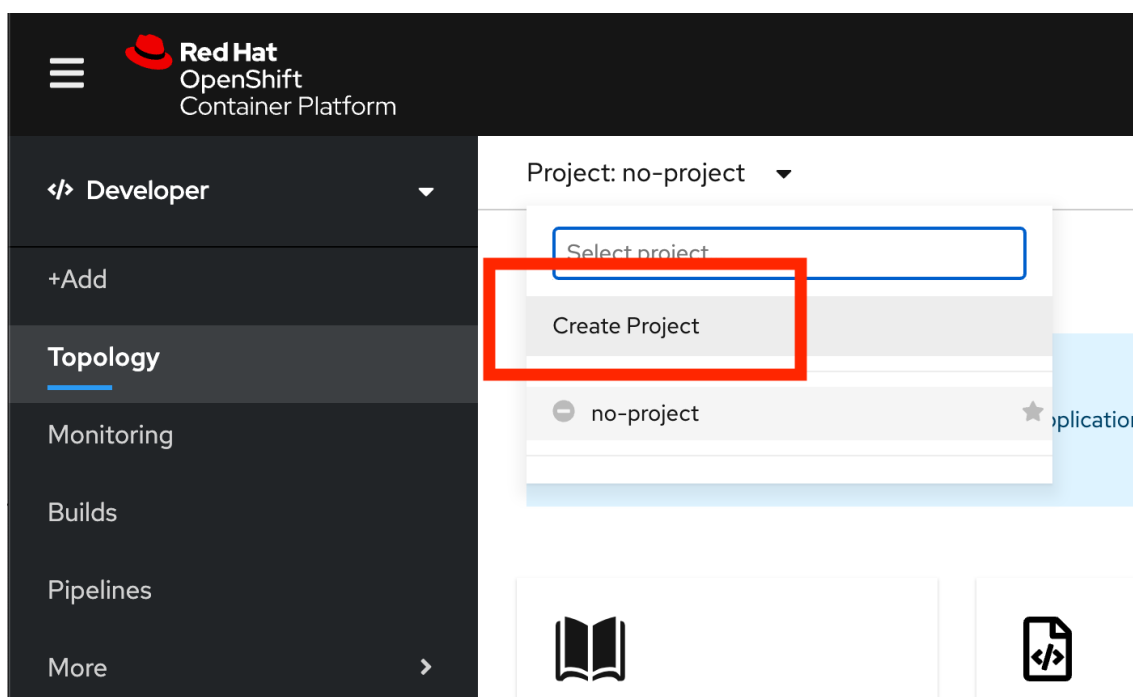
1. **解释** OpenShift容器平台web控制台提供了两个透视图;管理员透视图和开发人员透视图。Developer透视图提供了特定于开发人员用例的工作流。
2. **行动:** 使用透视图切换器切换到Developer透视图。显示带有创建应用程序选项的拓扑视图。



3.1. 展示项目创建

解释 项目允许用户社区独立于其他社区组织和管理其内容。项目是Kubernetes命名空间的OpenShift扩展，具有额外的功能来支持用户自我配置。在绝大多数情况下，它们是可以互换的。

1. **行动:** 单击项目下拉菜单，查看所有可用项目的列表。选择**创建项目**。



解释: 不同的项目如何拥有不同的用户权限和配额。

2. **行动:** 名称、显示名称和描述字段填写如下:

- **Name:** GUID-exploring-openshift

注意: 确保您将 GUID 替换为一个唯一的标识符，例如您的客户端名称或您从OPENTLC收到的四个字符的标识符。项目名称在OpenShift中必须是唯一的。

- **Display Name:** Exploring OpenShift UI

- **Description:** This is the project for exploring the openshift UI

Create Project

Name *

GUID-exploring-openshift

Display Name

Exploring OpenShift UI

Description

This is the project for exploring the OpenShift UI

Cancel Create

Browse the catalog to discover, Create resources from their YAML Browse

3.2. 演示应用程序部署

1. **解释:** web控制台中的Developer透视图为您提供以下添加视图的选项来创建应用程序和相关服务，并将它们部署到OpenShift容器平台上。
 - **解释:** 下面的过程将带领您在Developer透视图中使用**Import from git**选项来创建应用程序。使用GitHub中已有的代码库在OpenShift容器平台上创建、构建和部署一个应用，如下所示:
2. **行动:** 在Add视图中，点击**From Git**查看**Import From Git** form。


3. **行动:** 在Git部分，为您想要用于创建应用程序的代码库输入Git存储库URL。例如，输入这个示例Node.js应用程序<https://github.com/sclorg/nodejs-ex>的URL。然后验证URL。

Project: guid-exploring-openshift Application: all applications

Import from git

Git

Git Repo URL *















Validated

> [Show Advanced Git Options](#)

Builder

Builder Image *

 **Builder image(s) detected.**
Recommended builder images are represented by ★ icon.

 Perl	 PHP	 Nginx	 Modern Webapp	 Httpd	 .NET Core	 Go
 Ruby	 Python	 Java	 Node.js			

4. **解释:** 在Builder部分中，验证URL后，将检测到适当的构建器映像，由星号表示，并自动选择。对于 `https://github.com/sclorg/nodejs-ex` Git URL，默认选择Node.js构建器映像。如果需要，可以使用Builder Image version下拉列表更改版本。

5. **行动:** 在一般部分，执行以下操作:

- 在**Application**字段中输入唯一的应用分组名称，例如 "nodejs-ex-app"。确保应用程序名称在命名空间中是唯一的。
- 在 **name** 字段中输入唯一的名称，以标识为该应用程序创建的资源。这是根据Git存储库URL自动填充的。

提示: 资源名称在命名空间中不能重复。如果出现错误，请修改资源名称。

6. **行动:** 在 **资源** 中，选择默认资源 **部署**。

7. **解释:** 部署创建一个简单的Kubernetes风格的应用程序。“部署配置”创建一个OpenShift样式的应用程序。“Knative Service”创建一个微服务。

提示: Knative Service选项只有在集群中安装了Serverless Operator时才会显示在Import from git表单中。要了解更多信息，请参阅有关安装OpenShift Serverless的文档。

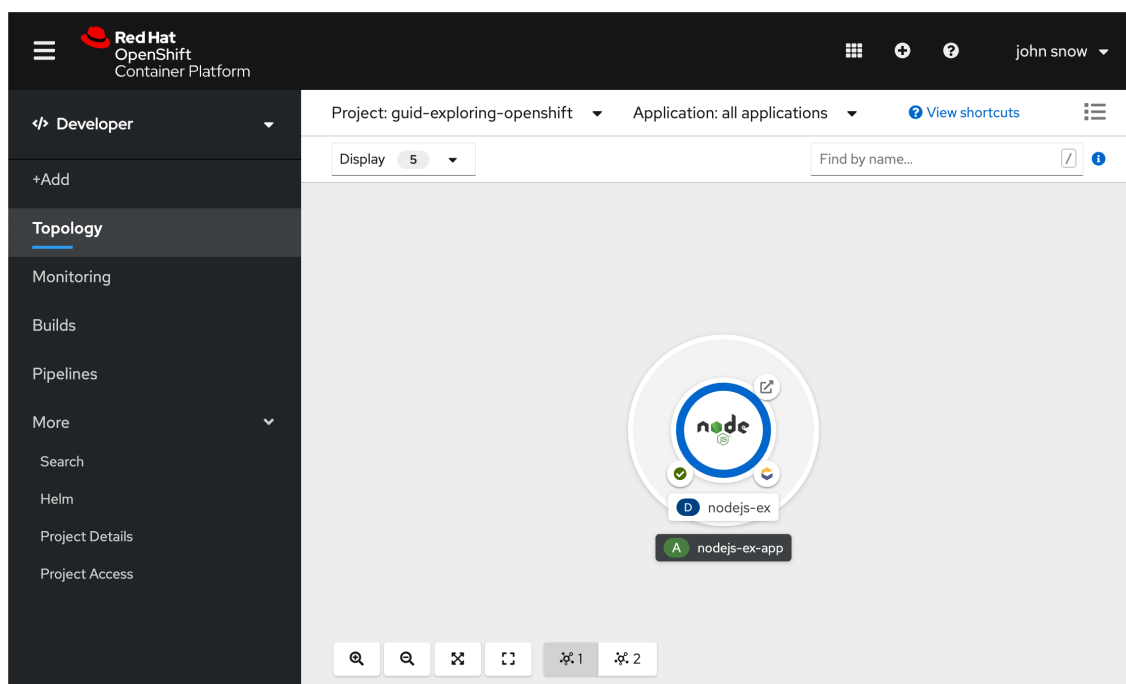
8. **解释:** 在**高级选项**部分，**创建一个路由到应用程序**默认选择，以便您可以使用一个公开可用的URL访问您的应用程序。如果不希望在公共路由上公开应用程序，可以清除此复选框。

9. **行动:** 单击**Create**创建应用程序，并在拓扑视图中查看其构建状态。

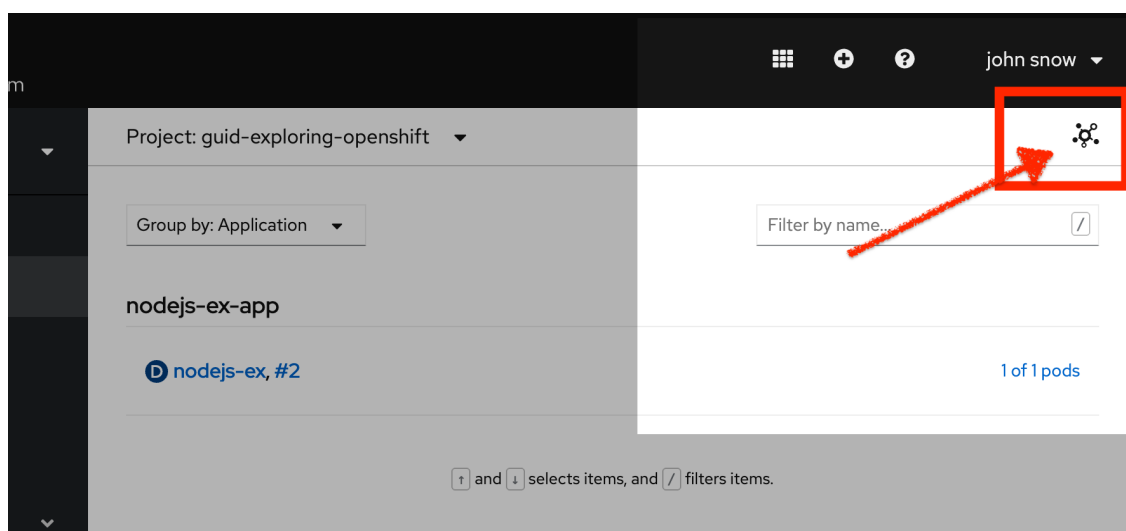
注意: 拓扑视图可能显示“1 Error”。这是正常的。等待构建完成，部署#2完成。您仍然可以期望您的应用程序能够成功地构建和部署。

3.3. 演示拓扑视图

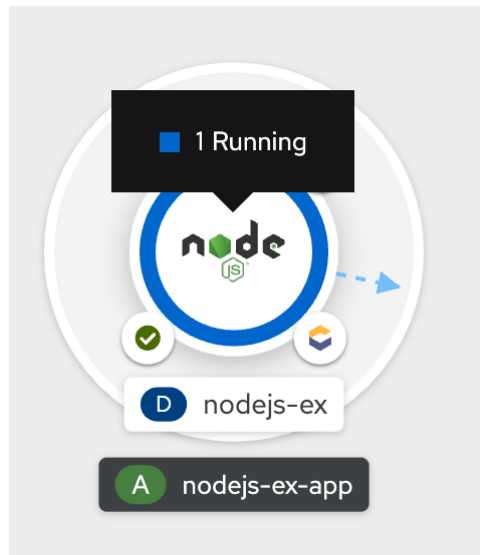
1. **解释:** web控制台的Developer透视图中的拓扑视图提供了项目中所有应用程序的可视化表示，它们的构建状态，以及与之相关的组件和服务。



行动: 如果没有图形化的表示方式，请单击web控制台右上角的“拓扑视图”图标。



2. **解释:** 您可以使用Developer透视图中的左侧导航面板导航到Topology视图。创建应用程序后，您将自动导向拓扑视图，在这里您可以看到应用程序Pods的状态，快速访问公共URL上的应用程序，访问源代码来修改它，并查看上一次构建的状态。您可以放大或缩小以查看特定应用程序的更多细节。
3. **解释:** Pod的状态或阶段由不同的颜色和工具提示表示，如Running (■), Not Ready (■), Warning (■), Failed (■), Pending (■), Succeeded (■), Terminating(■), or Unknown(■). 有关Pod状态的更多信息，请参阅Kubernetes文档。
4. **解释:** 创建应用程序并部署镜像后，状态显示为Pending。应用程序构建完成后，显示为Running。



- 应用程序资源名称(nodejs-ex)后面附加了不同类型资源对象的指示符，如下所示: **D**: Deployment **DC**: Deployment Configs **SS**: StatefulSet **DS**: Daemonset

注意：除了OpenShift部署配置，Kubernetes **Deployments**也被支持。Kubernetes部署共享Deployment Configs中的许多可用特性，并且是OpenShift容器平台4.5中的默认部署资源对象。

3.4. 演示与概览面板的交互

- 解释:** 有一个Overview面板，可以打开来访问Deployment的许多特性。

- 行动:** 单击**D nodejs-ex**打开概览面板。
 - nodejs-ex Deployment的详细信息可以在这里找到。这包括详细信息、资源和监视。
- 行动:** 单击概览面板的**Details**标签。

The screenshot shows the OpenShift console interface for the 'nodejs-ex' deployment. The top bar displays 'D nodejs-ex' and an 'Actions' dropdown menu. Below the bar, there are three tabs: 'Details', 'Resources', and 'Monitoring'. The 'Details' tab is active, showing a large blue circle with the number '1' and the text 'pod'. To the right of the circle are up and down arrows. Below the circle, there are several key-value pairs: 'Name: nodejs-ex', 'Update Strategy: RollingUpdate', 'Namespace: NS guid-exploring-openshift', 'Max Unavailable: 25% of 1 pod', 'Labels: app=nodejs-ex, app.kubernetes.io/comp... =nodejs...', and 'Max Surge: 25% greater than 1 pod'.

- 解释** 有以下几个方面内容：
 - 可以在这里管理Pod复制计数。您可以使用向上和向下箭头扩展pod，以手动增加或减少应用程序的实例数量。对于无服务器应用程序，pod在空闲时自动缩小到零，然后根据通道流量进行扩展。

- Pod配置和可点击上下文帮助的几个方面。查看应用程序的“标签”、“标注”和“状态”。

2. **行动:** 单击资源页签。

The screenshot shows the OpenShift console interface for a deployment named 'nodejs-ex'. The 'Resources' tab is selected, displaying a list of associated resources. On the left, a sidebar shows the deployment's icon and name. The main content area is divided into sections: 'Pods' (showing one running pod), 'Builds' (showing one completed build), 'Services' (showing one service), and 'Routes' (showing one route). Each resource entry includes its name, icon, and a 'View logs' link. The 'Start Build' button is visible next to the build entry.

- **解释** 有以下几个方面内容：
 - 部署创建了若干**资源**，它们的状态在这里显示。
 - 这些只是与部署关联的一些资源。
 - **pod**是OpenShift应用程序的基本单元，可以访问它们的日志。
 - 用于编译源代码并将其打包成镜像。查看它们的状态、访问日志，并在需要时启动新的构建。
 - **Services** 已为您的Pod创建，并分配的端口被列出。
 - **Routes** 是为了允许外部访问pod，并列出了访问它们的URL。

3. **行动:** 点击review面板的**Monitoring**标签。

- 出现在这个页面上的各种**事件**和**度量**信息。

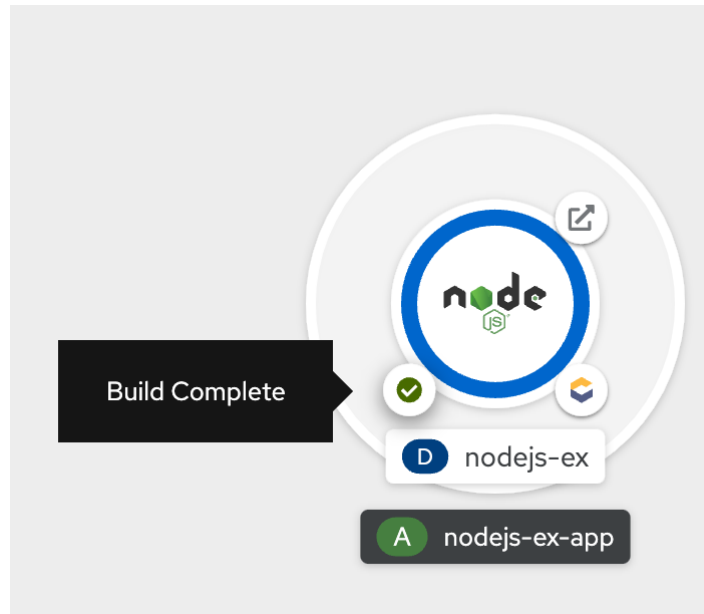
4. **行动:** 通过单击拓扑视图字段中的**X**或任何位置来关闭review面板。

提示: 对于无服务器应用程序，Resources选项卡提供了关于修订、路由和该组件使用的配置的信息。

3.5. 演示如何与应用程序和组件交互

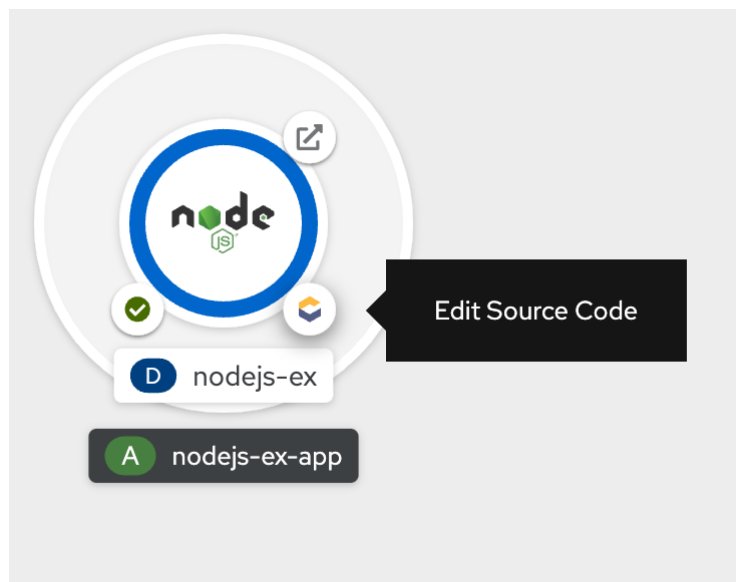
解释: web控制台的开发人员视图中的拓扑视图提供了以下选项来与应用程序和组件交互:

1. **行动:** 将鼠标悬停在Pod的左下方图标上, 可以看到最新版本名称及其状态。
 - Pod的构建状态指示器:

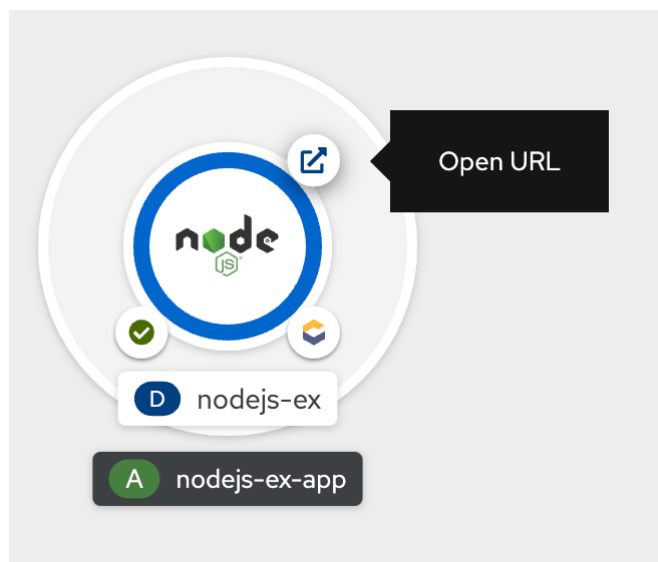


- **解释:** 应用程序构建的状态指示为新建、挂起、运行、完成、失败或取消。
2. **行动:** 将鼠标悬停在Pod右下角的图标上, 可以查看对源代码的访问。
 - **解释:** 您可以在“Code Ready Workspaces”中打开源代码并编辑应用程序代码。

提示: 只有当你使用From Git, From Catalog和From Dockerfile选项创建应用程序时, 这个特性才可用。

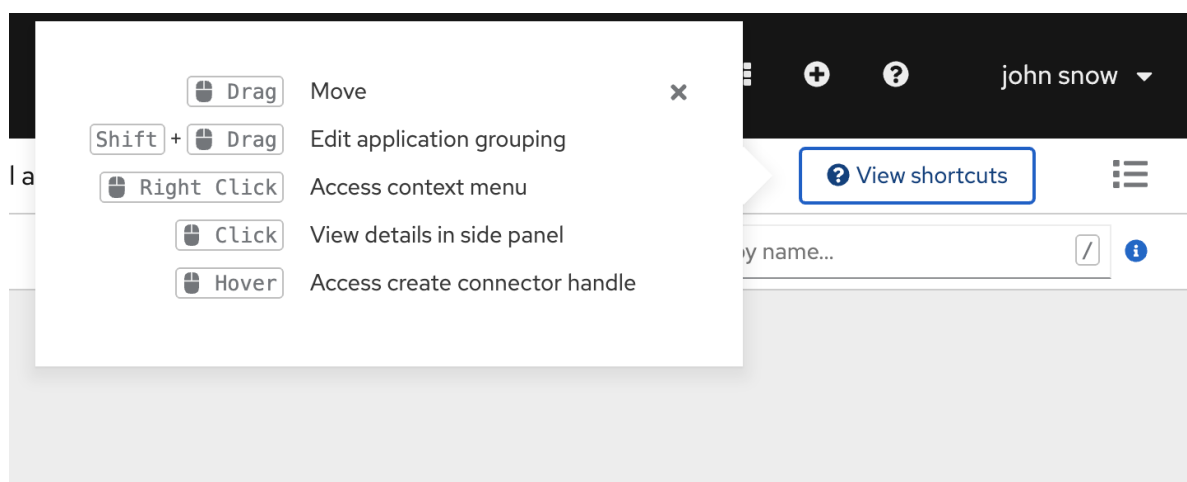


3. **行动:** 将鼠标悬停在右上角的图标上可以看到公共URL。
 - 应用程序URL是可用的:



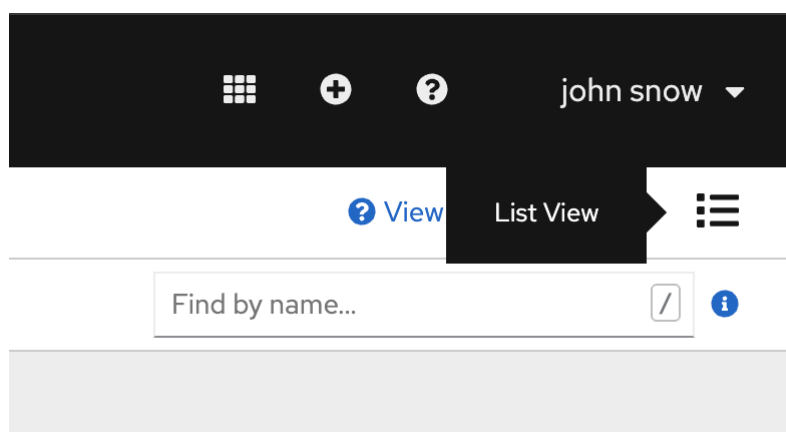
4. **解释:** 为了更容易地在拓扑视图中查看许多应用程序，有键盘和鼠标快捷键。

- **行动:** 使用屏幕右上方列出的快捷键菜单在Topology视图中导航组件。



5. **解释:** 有时，您可能希望在列表视图中查看所有应用程序及其资源。

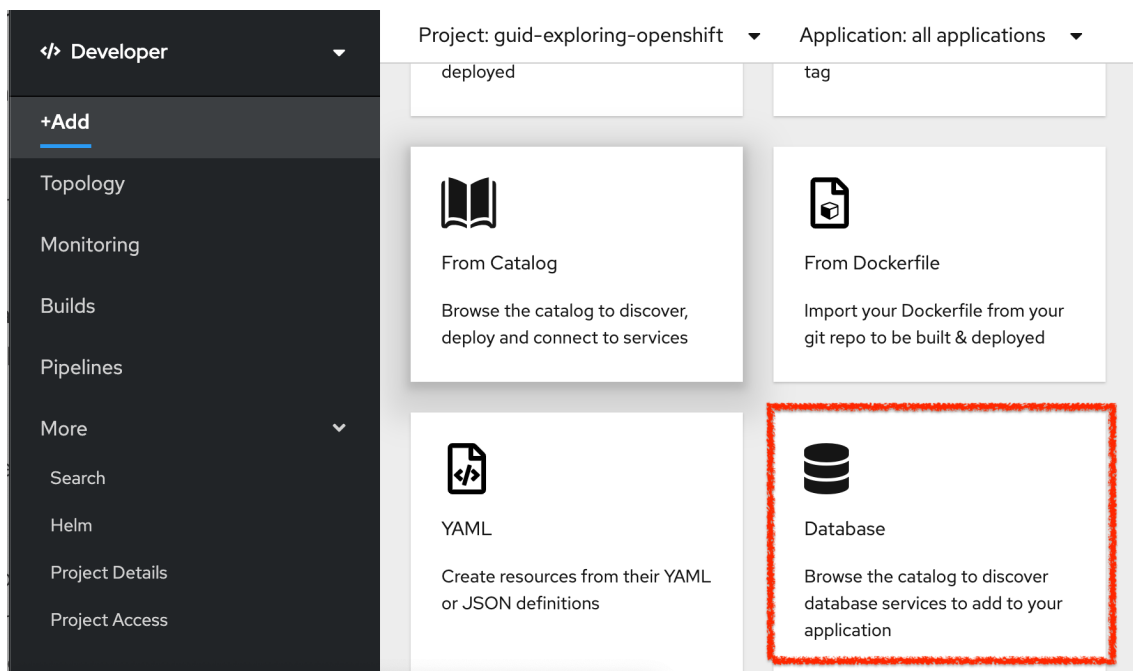
- **行动:** 使用“列表视图”图标查看所有应用程序的列表，并使用“拓扑视图”图标切换回“拓扑视图”。



3.6. 演示如何在应用程序中分组多个组件

解释: 可以使用“添加”页向项目添加多个组件或服务，并使用“拓扑”页在应用程序组中对应用程序和资源进行分组。下面的过程使用Node.js组件将MongoDB数据库服务添加到现有的应用程序中。

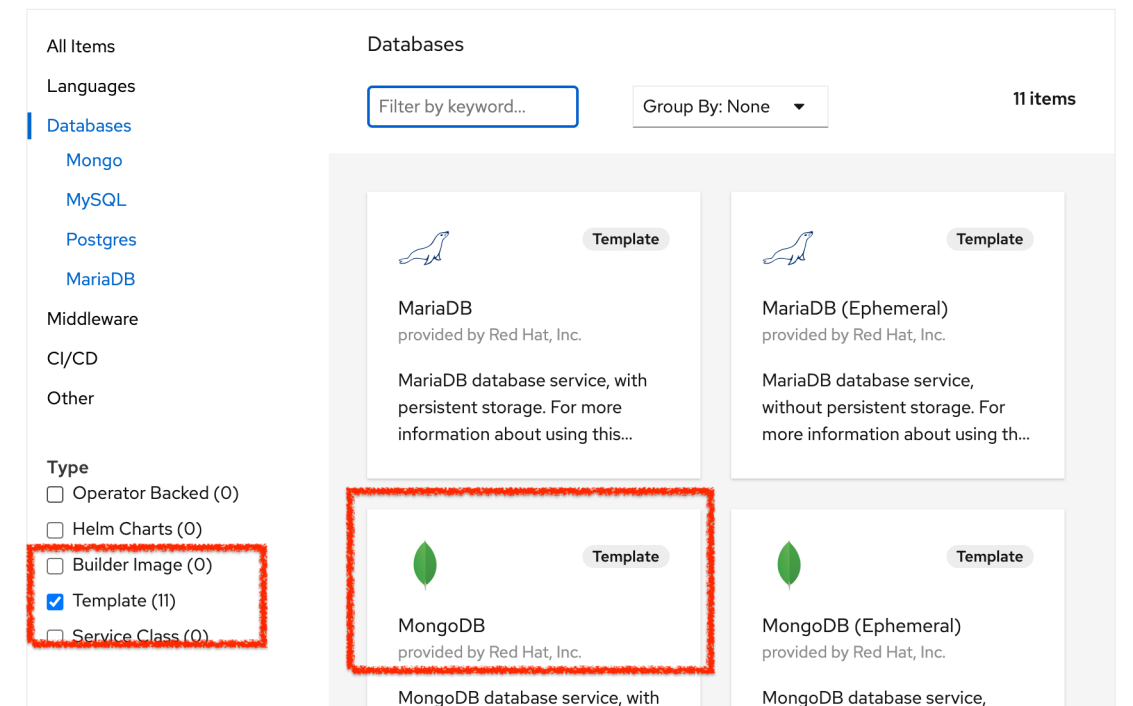
1. **行动:** 导航到Add视图并选择**Database**选项来查看开发人员目录。



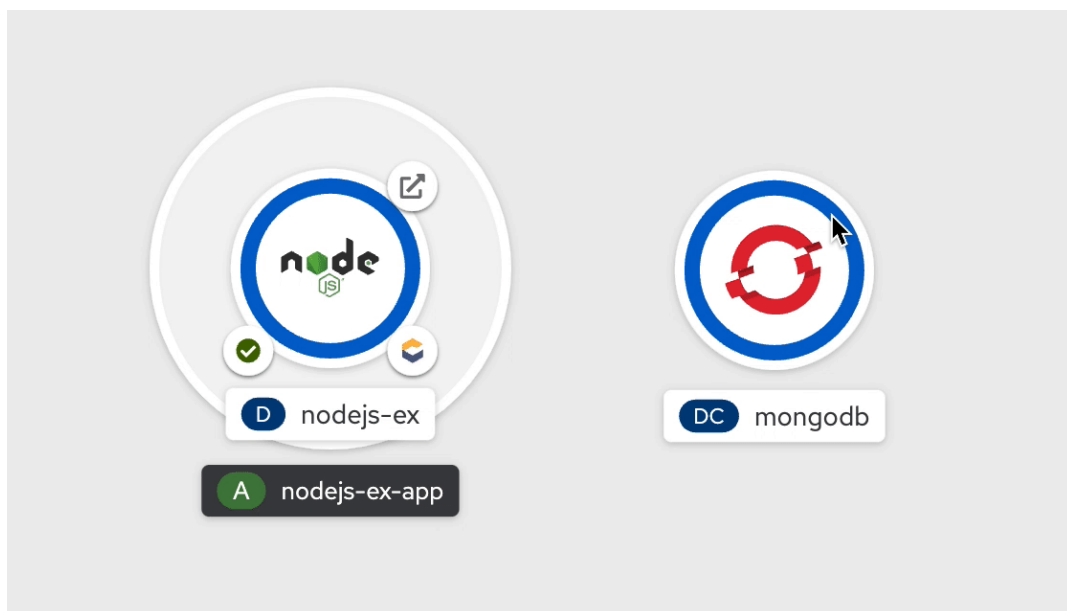
- **解释:** 目录有多个选项，您可以将它们作为组件或服务添加到应用程序中。
2. **行动:** 选择**Type: Template**复选框来显示基于OpenShift模板的目录项。

Developer Catalog

Add shared apps, services, or source-to-image builders to your project from the Developer Catalog. Cluster admins can install additional apps which will show up here automatically.



- **行动:** 单击 **MongoDB**(非MongoDB Ephemeral) 选项查看服务的详细信息。
 - **行动:** 单击**Instantiate Template**查看MongoDB服务的详细信息自动填充的模板。
 - **行动:** 滚动到页面底部，单击**Create**创建服务。
 - **行动:** 您应该自动跳转到拓扑视图，以观察MongoDB的部署情况。注意Pod圆是如何改变颜色来表示状态的。将鼠标悬停在Pod上查看状态。
3. **行动:** 在左侧导航面板中，单击**Topology**查看项目中部署的MongoDB服务。
- **行动:** 要将MongoDB服务添加到现有的应用组中，按住**SHIFT**，选择'MongoDB'Pod并拖动到应用中;完成MongoDB服务添加到已有应用组。



- 应用程序分组(nodejs-ex-app)被标记为A的扁圆形。
- 解释:** 按住SHIFT键拖动组件并将其添加到应用程序组中，将自动向组件添加所需的标签。

4. **行动:** 单击MongoDB **DC MongoDB** 进入Overview 面板，单击**Details**页签。

DC mongodb	
Details Resources Monitoring	
1 pod	
Name	mongodb
Latest Version	1
Namespace	NS guid-exploring-openshift
Message	config change
Labels	<div>app.kubernetes.io/p...=nodejs-ex...</div> <div>templ... mongodb-persistent-te...</div> <div>template.opens...=269d6c9f-f2b...</div>
Update Strategy	Recreate
Min Ready Seconds	Not Configured

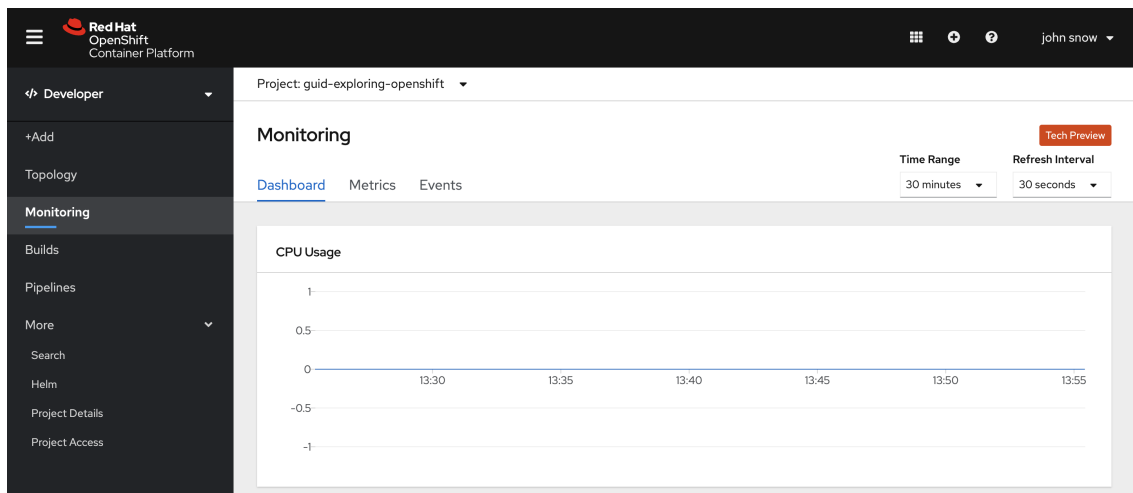
- 显示标签 `app.kubernetes.io/part-of=nodejs-ex` 被添加到标签部分。

提示: 要集成应用程序和数据库，Node.js应用程序和MongoDB数据库需要在其他地方配置。在OpenShift的未来版本中，可以看到基于Operator的服务绑定。

3.7. 演示在开发人员的视角中探索监控页面

解释: 红帽公司最近将监控功能集成到网络控制台中。在这里探索项目范围的度量衡和事件。

1. **行动:** 在左侧面板，点击**监控**:



○ **解释** 如下:

- 这是一个技术预览功能，取代依赖于Grafana。
- 这些特性只适用于上面所选的项目。

2. **行动:** 点击**Dashboard**标签页:

- Dashboard, 它显示了项目的组合指标。

3. **行动:** 点击**Metrics**标签页:

- Metrics选项卡, 在这里可以创建Prometheus Metrics的自定义图形。

4. **行动:** 点击“**事件**”页签:

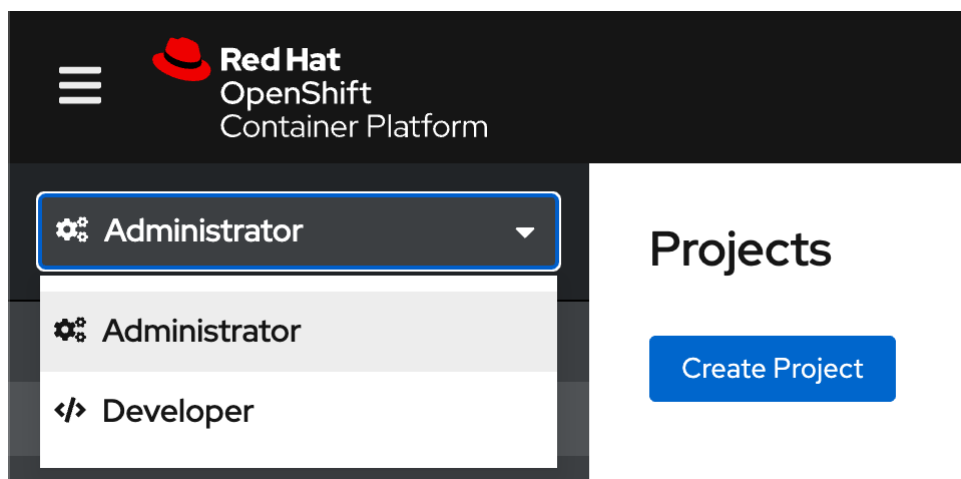
- Events选项卡, 其中的事件被报告为一个流, 并且可以过滤。

4. 演示部署和部署配置

4.1. 演示切换到管理员透视图

1. **解释:** 许多高级功能只能通过管理员透视图使用。

- **行动:** 在透视图下拉框中, 选择**管理员**:



5. 演示部署和部署配置

1. **解释:** OpenShift容器平台中的`Deployment`和`Deployment Configs`是API对象, 它们提供了两种类似但不同的方法来细粒度管理普通用户应用程序。它们由以下独立的API对象组成:

- Deployment Config 和 Deployment的其中任何一个都将应用程序的特定组件的期望状态描述为Pod模板。
- Deployment Configs包括一个或多个replicationcontroller, 其中包含一个Deployment Config状态的时间点记录作为Pod模板。类似地, Deployments包括一个或多个replicaset, 它是ReplicationControllers的后继者。

- 一个或多个Pods，它们表示应用程序特定版本的实例。

2. 行动: 从左侧导航中，单击工作负载→部署配置

- 从每个Deployment Config右侧的选择菜单中，您可以启动所需的Deployment Config的新Rollout。

Deployment Configs

Create Deployment Config					Filter by name...
Name ↑	Namespace ↓	Status ↓	Labels ↓	Pod Selector ↓	
DC mongodb	NS guid-exploring-openshift	1 of 1 pods	app.kubernetes.io=name=nodejs-... tem...=mongodb-persist... template.o...=269d6c9f...	Q name=mongodb	<div>Start Rollout Pause Rollouts Edit Count ...</div>

3. 行动: 从列表中单击 mongodb 部署配置，然后打开部署配置详细信息页面。

Project: guid-exploring-openshift

Deployment Configs > Deployment Config Details

DC mongodb

Actions

DetailsYAMLReplication ControllersPodsEnvironmentEvents

Deployment Config Details

1pod

Namemongodb

NamespaceNS guid-exploring-openshift

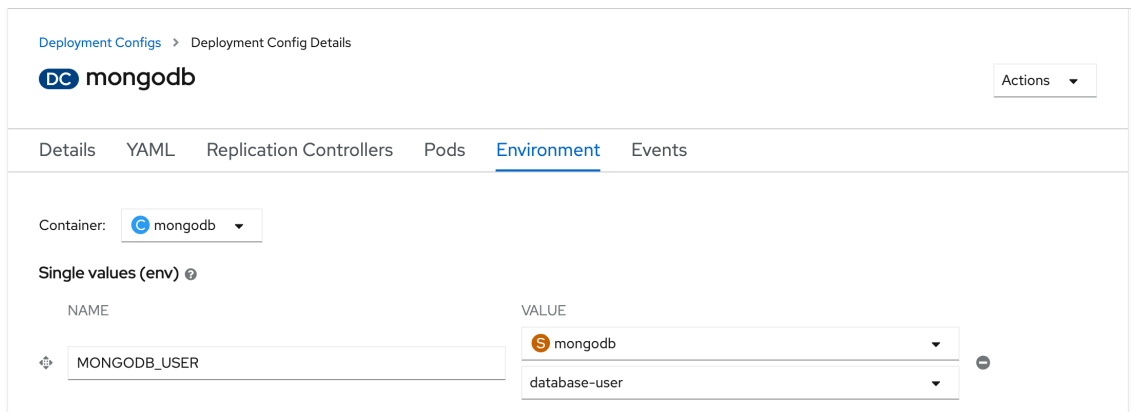
Latest Version1

Messageconfig change

- 解释 如下：

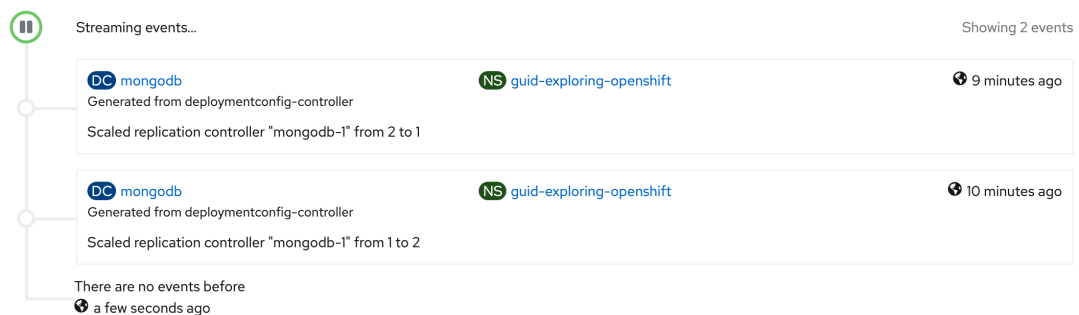
- 在圆上指出pod部署的状态。
- 从这个面板上点击圆圈旁边的上下箭头可以很容易地增加或减少Pod的所需数量 (即 replicas)。
- 你可以看到以下内容：
 - 部署配置的**最新版本**
 - 最新**消息**显示为什么当前部署启动
 - 页面下面是**Containers**的列表，它们的起始镜像、资源限制和它们监听的端口
 - **条件**列表，其中表达了与此部署配置相关的主要事件，包括
NewReplicationControllerAvailable **原因**和 replication controller "mongodb-1" successfully rolled out **消息**
- 您也可以通过单击右上角的**Actions**菜单并选择**start Rollout**来启动一个新的部署，这还会创建一个新的replication controller。

4. 行动: 选择环境页签。



- **解释:** 环境选项卡显示为Deployment设置的环境变量。
 - 环境变量用于设置容器和pod中的不同参数，例如用户名、数据库服务名等。
 - 您可以添加自己的环境变量，并从应用程序的代码中处理它们。

5. **行动:** 选择**Events**页签。



- **解释** 在**Events**选项卡中，您可以看到与您的部署相关的事件。
 - 事件列表提供了一种有用的方法，可以查看部署中是否出现了错误，或者回溯事件链。

6. **行动:** 在左侧选择**Workload**→**Replication Controllers**。

- **解释** Deployment Configs 创建 *replication controllers*。

7. **行动:** 点击replication controller的名称 `mongodb-1`，打开**Replication Controller Overview**页面。

RC mongodb-1 Complete[Details](#) [YAML](#) [Pods](#) [Environment](#) [Events](#)

Replication Controller Details

Name

mongodb-1

Phase

Complete

Namespace

NS guid-exploring-openshift

Current Count

1

Labels

openshift.io/deployment-config.name=mongodb

template=mongodb-persistent-template

template.openshift.io/template-instance-id=269d6c9f-f2bb-42a8-b82f-fc8d1...

Desired Count

1

Pod Selector

deployment=mongodb-1, deploymentconfig=mongodb, name=mongodb

解释如下:

- `mongodb-1` replication controller有一个Pod选择器, 它表示部署的版本和创建该部署的部署配置的名称。
- Replication controller按照Deployment Config的要求管理实际推出的pod的数量。

提示: 解释Replicationcontroller与replicaset之间的关系。它类似于OpenShift Deployment Configs 和Kubernetes Deployment 之间的关系。所有这些都由OpenShift支持。

5.1. 演示用于部署的网络资源

1. 行动: 在左侧菜单中选择Networking→Services:

- **解释** 项目的Services页面显示了项目中当前存在的所有服务。
- **Location**是服务用来表示其Pods的永久内部IP地址和端口。

2. 行动: 点击mongodb服务:

Services > Service Details

S mongodb Actions

[Details](#) [YAML](#) [Pods](#)

Service Details

Name
mongodb

Namespace
NS guid-exploring-openshift

Labels
template=mongodb-persistent-template
template.openshift.io/template-instance-id=269d6c9f-f2bb-42a8-b82f-fc8d13...

Pod Selector
name=mongodb

Annotations
1 Annotation

Service Routing

Service Address





Type	Location
Cluster IP	172.30.152.29
Accessible within the cluster only	

Service Port Mapping

Name	Port	Protocol	Pod Port or Name
mongo	27017	TCP	27017

- 服务正在监听的服务地址和服务端口映射是前面屏幕中的位置，但显示了服务端口映射的Pod的端口。
 - **Pod Selector**描述了须将Pod视为服务的一部分的标签。
3. **行动:** 在左侧菜单中选择**Networking**→**路由**:
- **解释** 项目的**Routes**页面显示了项目中当前存在的所有路由。
 - 路由列表，它们的位置，以及它们所代表的服务。
 - 标记为 `nodejs-ex` 的路由信息行。
 - 有一个路由 `nodejs-ex`，它公开一个服务，也称为 `nodejs-ex`。
 - **Route**暴露的URL。

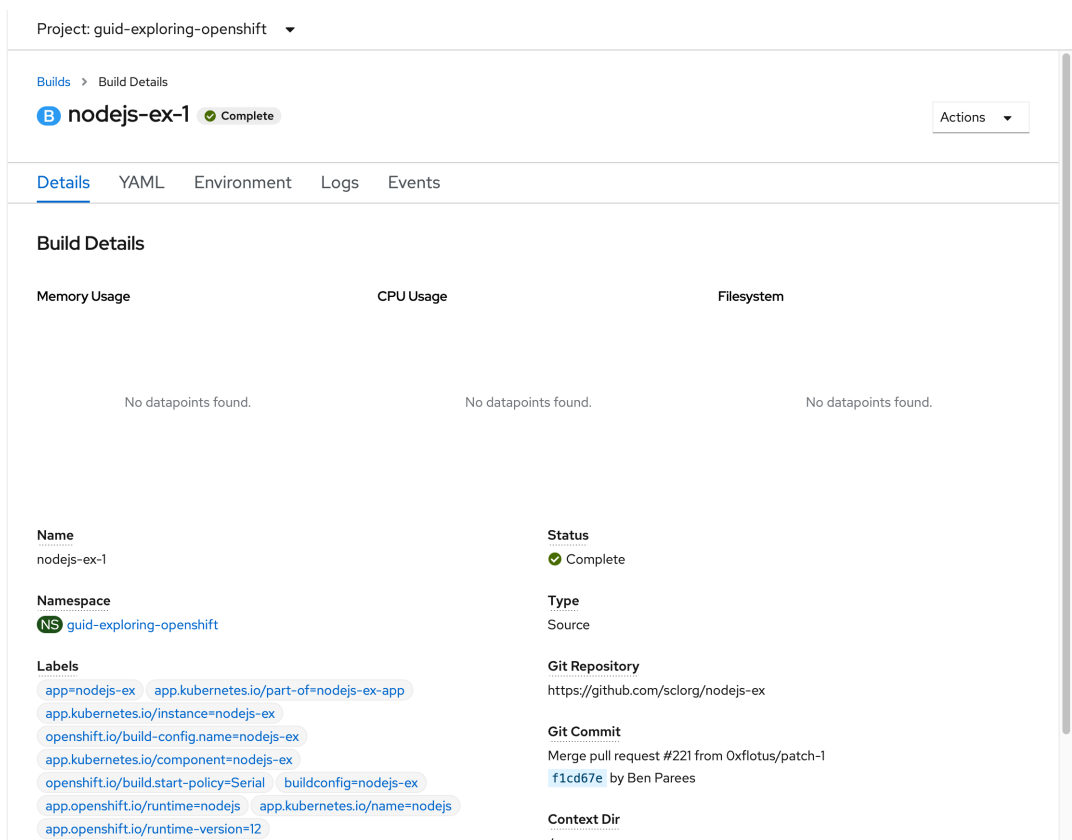
Routes

Create Route		Filter by name...		
2 Accepted	0 Rejected	0 Pending	Select all filters	2 Items
Name ↑	Namespace ↓	Status	Location ↓	Service ↓
 nodejs-ex	 guid-exploring-openshift	 Accepted	http://nodejs-ex-guid-exploring-openshift.apps.shared-na4.na4.openshift.opentlc.com	 nodejs-ex

6. 演示探索构建页面

在本节中，您将查看UI中关于构建的可用信息。

1. **行动:** 从左边的菜单中选择**Builds**→**Build Configs**:
 - **解释** **Builds Configs**页面显示了项目的所有BuildConfigs。
2. **行动:** 从列表中单击 `nodejs-ex` BuildConfig， **Build Config Details**页面打开
 - 可以通过点击右上角的**Actions**菜单并选择**start build**来开始一个新的构建。
 - 您可以看到**GIT REPOSITORY**，其中存储了构建的源代码。
3. **行动:** 选择**Builds**选项卡，将打开构建页面列表。
 - **解释** 项目的 **构建**页面显示项目的所有构建。
 - 您可以看到项目中每个构建的状态和完成时间。
4. **行动:** 在列表选择一个版本:
 - **解释** 在**application**的 **Build Details**页面中，你可以看到以下内容:



- 您可以在**Details**选项卡中查看此构建使用的配置。
- 构建使用的资源利用统计数据如果可用，将以图形形式显示在此页面上。
- 你可以看到构建的**所有者**。在本例中，所有者是 `nodejs-ex` BuildConfig，您在前面的步骤中看到了它。
- 请注意，您可以通过单击右上角的**Actions**菜单并选择**rebuild**来重新生成此版本。
- 您可以看到构建的状态和触发它的原因。
- 您可以看到构建的配置，包括所使用的基本镜像和输出镜像的名称。

5. **行动:** 选择**Logs**页签。

- **解释** 在**Logs**页签中显示构建的日志。
- **解释** 在其他示例中，您可能会看到容器和其他配置日志下拉的代码依赖项。
 - 在本例中，源代码存储库被克隆或下载到容器中。
 - 您可以看到已完成的镜像被自动推送到您的项目名称下的集成注册表中。

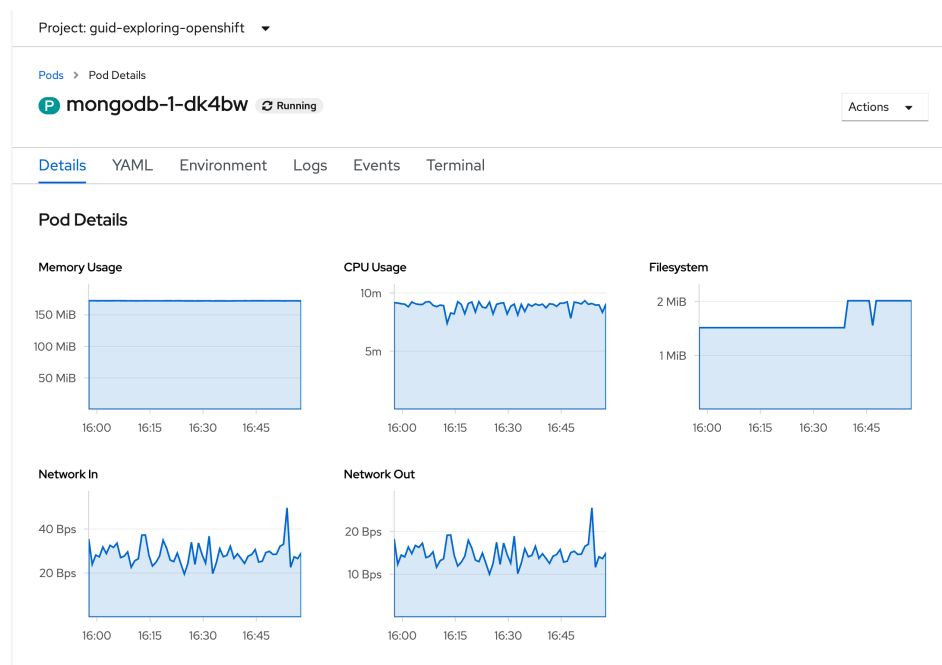
7. 演示探索Pods

1. **行动:** 在左侧菜单中选择**Workloads**→**Pods**:

- **解释** 项目的**Pods**页面显示了项目中当前运行的所有Pods。
 - 你可以看到容器的状态: `Running`, `Pending`, 等等。
 - **Ready**列显示了基于就绪检查的容器中应用程序的真实状态。

2. **行动:** 当你完成后，从列表中选择你的'mongodb-XXXXX'Pod。

- **解释** **Pod**页面显示如下:
 - 在**Pod Overview**中:
 - 展示**内存使用率**，**CPU使用率**和**文件系统**的图表。



- Pod的状态和托管它的OpenShift节点。
 - 在**Containers**部分中显示的信息，包括镜像名称和容器状态。
3. **行动:** 点击名为 `mongodb` 的容器，**容器详细信息**页面打开。
- **解释** 如下:
 - 你的Node.js应用程序容器的**资源请求和资源限制**。
 - PHP应用程序容器的 **Readiness** 和 **Liveness** 探针。
4. **行动:** 点击浏览器上的**Back**，进入**Pod详细信息**页面。
- **解释** 如下:
 - 在**Mounted Volumes**中显示的信息，包括卷的名称、类型和权限。
 - 目前有挂载两个卷:
 - `mongodb-data`卷绑定到PVC **mongodb-data**，具有读写权限
 - `default-token` **Secret** 卷已绑定，为**Read-only**。
5. **行动:** 完成后，选择**Logs**选项卡。
- **解释** 在**Logs**页签中显示Pod的日志:
 - 此处显示Pod消息，并注意您可以在日志更新时跟踪日志。
 - 您可以在日志更新时暂停并重新启动日志流。
6. **行动:** 完成后，选择**Terminal**选项卡。
- **解释** **Terminal**选项卡允许您在Pod中的任何容器内使用一个终端。
 - 您可以在容器中运行命令进行调试和测试。
7. **行动:** 当您完成后，选择**Events**选项卡。
- **解释** **Events**页签显示与Pod相关的事件。
 - 您可以使用这个列表来查看Pod部署中是否出现了错误，或者追溯事件链。

8. 清理环境

如果您不打算做任何额外的演示，请转到OpenShift容器平台web控制台的主页并删除您的项目。

