OpenShift Container Platform 3.11

开始使用

开始使用 OpenShift Container Platform 3.11

Last Updated: 2020-09-25

OpenShift Container Platform 3.11 开始使用

开始使用 OpenShift Container Platform 3.11

法律通告

摘要

无论您是开发人员还是平台管理员，都可以使用本书中介绍的内容来开始使用 OpenShift。管理员可以使用安装实用程序和交互式 CLI 工具在多个主机上快速安装和配置 OpenShift 实例。开发人员可以使用 OpenShift CLI 或 Web 控制台登录到现有的 OpenShift 实例并开始创建应用程序。

目录

第 1 章 概述

1.1. 简介

OpenShift Container Platform 将 Docker 和 Kubernetes 整合在一起，提供用于管理这些服务的 API。OpenShift Container Platform 允许您创建和管理容器。

1.1.1. 为什么我应该使用 OpenShift？

容器是独立于操作系统和底层基础架构的、在其自己的环境中运行的独立进程。OpenShift 可以帮助您开发、部署和管理基于容器的应用程序。它为您提供了一个自助服务平台，供您根据需要创建、修改和部署应用程序，从而可以更快地进行开发和发行生命周期。

您可以将镜像（image）*想象为*饼干模具，而容器（container）是实际的饼干。

您可以将 OpenShift *看作为*一个操作系统，镜像为在其上您运行的应用程序，而容器是这些应用程序的实际运行实例（instance）。

根据您的角色查找合适的主题以开始：

我是一个	链接到相关主题
平台管理员	安装基本 OpenShift Container Platform 环境或 安装生产环境中的 OpenShift Container Platform 环境。
开发者	使用 Web 控制台来步骤如何创建和构建镜像的基础知识， 并创建第一个项目和应用程序。

第 2 章 安装 OPENSHIFT CONTAINER PLATFORM

2.1. 概述

本指南介绍 OpenShift Container Platform 的基本概念，并帮助您安装基本应用程序。本指南不适用于在生产环境中部署或安装 OpenShift Container Platform。

2.1.1. 先决条件

要安装 OpenShift Container Platform，您需要具备以下条件：

- 至少两台物理的或虚拟的 RHEL 7+ 机器，具有完全限定的域名（可以是真实的域名，也可以是网络中的域名）和 无密码的 SSH 访问权限。本指南使用 **master.openshift.example.com** 和 **node.openshift.example.com**。这些机器必须能够使用这些域名相互 ping 通。如果您使用 IBM POWER 服务器，您的集群中的所有服务器都必须是 IBM POWER 服务器。
- 有效的红帽订阅。
- 可以将您的域解析到节点 IP 的通配符 DNS 解析条目。因此，您的 DNS 服务器中应该有与以下类似的条目：

```
master.openshift.example.com. 300 IN A <master_ip>
node.openshift.example.com. 300 IN A <node_ip>
*.apps.openshift.example.com. 300 IN A <node_ip>
```

为什么您的域名中使用 APPS 作为通配符条目？

当使用 OpenShift Container Platform 部署应用程序时，内部路由器需要将传入请求代理到对应的应用程序 pod。通过将 **apps** 用作应用程序域的一部分，应用程序流量会准确标记为正确的 pod。

您也可以使用 **apps** 以外的其他项。

```
*.cloudapps.openshift.example.com. 300 IN A <node_ip>
```

配置完成后，请按照以下步骤设置双机 OpenShift Container Platform 安装。

2.1.2. 附加 OpenShift Container Platform 订阅

1. 作为 root 用户在目标机器（包括 master 和节点）上，使用 **subscription-manager** 在红帽注册系统。

```
# subscription-manager register
```

2. 从 Red Hat Subscription Manager(RHSM)获取最新的订阅数据：

```
# subscription-manager refresh
```

3. 列出可用的订阅：

```
# subscription-manager list --available
```

4. 查找提供和附加 OpenShift Container Platform 订阅的池 ID。

```
# subscription-manager attach --pool=<pool_id>
```

5. 使用提供了 OpenShift Container Platform 的池的池 ID 替换字符串 **<pool_id>**。池 ID 是一个长字母数字字符串。

现在，这些 RHEL 系统被授权安装 OpenShift Container Platform。现在，您需要告诉系统从哪里获取 OpenShift Container Platform。

2.1.3. 设置软件仓库

在 master 和节点上，使用 **subscription-manager** 启用安装 OpenShift Container Platform 所需的软件仓库（repository）。在这个示例中，您可能已经启用了前两个库。

- 对于 x86_64 服务器中的云安装和内部安装，请运行以下命令：

```
# subscription-manager repos --enable="rhel-7-server-rpms" \
    --enable="rhel-7-server-extras-rpms" \
    --enable="rhel-7-server-ose-3.11-rpms" \
    --enable="rhel-7-server-ansible-2.9-rpms"
```

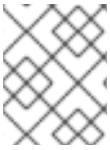
- 对于 IBM POWER8 服务器中的内部安装，请运行以下命令：

```
# subscription-manager repos \
    --enable="rhel-7-for-power-le-rpms" \
    --enable="rhel-7-for-power-le-extras-rpms" \
    --enable="rhel-7-for-power-le-optional-rpms" \
    --enable="rhel-7-server-ansible-2.9-for-power-le-rpms" \
    --enable="rhel-7-server-for-power-le-rhscl-rpms" \
    --enable="rhel-7-for-power-le-ose-3.11-rpms"
```

- 对于 IBM POWER9 服务器中的内部安装，请运行以下命令：

```
# subscription-manager repos \
    --enable="rhel-7-for-power-9-rpms" \
    --enable="rhel-7-for-power-9-extras-rpms" \
    --enable="rhel-7-for-power-9-optional-rpms" \
    --enable="rhel-7-server-ansible-2.9-for-power-9-rpms" \
    --enable="rhel-7-server-for-power-9-rhscl-rpms" \
    --enable="rhel-7-for-power-9-ose-3.11-rpms"
```

此命令告知您的 RHEL 系统，安装 OpenShift Container Platform 所需的工具可从这些软件仓库中获得。现在，我们需要基于 Ansible 的 OpenShift Container Platform 安装程序。



注意

旧版本的 OpenShift Container Platform 3.11 仅支持 Ansible 2.6。Playbook 的最新版本现在支持 Ansible 2.9，这是首选的版本。

2.1.4. 安装 OpenShift Container Platform 软件包

OpenShift Container Platform 的安装程序由 **openshift-ansible** 软件包提供。在运行了 **yum update** 后，在 master 上使用 **yum** 进行安装。

```
# yum -y install wget git net-tools bind-utils iptables-services bridge-utils bash-completion kexec-tools
sos psacct
# yum -y update
# reboot

# yum -y install openshift-ansible
```

现在安装一个容器引擎：

- 安装 CRI-O:

  ```
  # yum -y install cri-o
  ```

- 安装 Docker:

  ```
  # yum -y install docker
  ```

2.1.5. 设置无密码 SSH 访问

在 master 上运行安装程序前，请设置无密码 SSH 访问权限，因为安装程序需要它才能访问该机器。为此，请运行以下命令：

```
$ ssh-keygen
```

按照提示操作，并在被要求输入 pass phrase 时按回车键。

分发 SSH 密钥的简便方法是使用 **bash** loop：

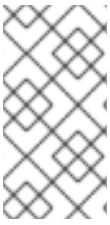
```
$ for host in master.openshift.example.com \
    node.openshift.example.com; \
    do ssh-copy-id -i ~/.ssh/id_rsa.pub $host; \
    done
```

2.1.6. 运行 Installation Playbook

1. 请参阅 清单文件实例，并选择最接近您所需集群配置的示例。

重要

红帽不支持使用 **--tags** 或 **--check** 选项运行 Ansible playbook。



注意

另外，在 */usr/share/doc/openshift-ansible-docs-3.11.<version>/docs/example-inventories/* 目录（用您最新安装的 **openshift-ansible-docs** 软件包版本替换 **<version>**，这将随着 **openshift-ansible** 父软件包升级而更新）。如需有关可用清单变量的完整文档，请参阅配置您的清单文件。

1. 编辑示例清单，使用您的主机名，然后将其保存到文件中。默认位置是 */etc/ansible/hosts*。

2. 进入 playbook 目录，并使用您的清单文件运行***prerequisites.yml*** playbook：

```
$ cd /usr/share/ansible/openshift-ansible
$ ansible-playbook -i <inventory_file> playbooks/prerequisites.yml
```

3. 进入 playbook 目录，并使用您的清单文件运行 ***deploy_cluster.yml*** playbook:

```
$ cd /usr/share/ansible/openshift-ansible
$ ansible-playbook -i <inventory_file> playbooks/deploy_cluster.yml
```

在安装成功后，但在添加新项目前，您必须设置基本身份验证、用户访问和路由。

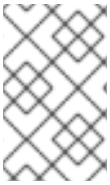
2.2. 与 OPENSHIFT CONTAINER PLATFORM 进行交互

OpenShift Container Platform 提供了两个命令行实用程序来与它交互。

- **oc**: 用于常规的项目和应用程序管理
- **oc adm**: 用于执行系统管理任务
  运行 **oc adm** 命令时，您应该只从 Ansible 主机清单文件中列出的第一个 master 运行它们，默认为 ***/etc/ansible/hosts***。

使用 **oc --help** 和 **oc adm --help** 查看所有可用选项。

另外，您可以使用 Web 控制台来管理项目和应用程序。Web 控制台位于 **https://<master_fqdn>:8443/console**。在下一部分，您可以看到如何创建访问控制台的用户帐户。



注意

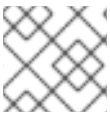
也可以使用这些命令行实用程序，与远程系统 OpenShift Container Platform 实例交互。作为捆绑的 OpenShift CLI，您可以参照 CLI 参考中的内容下载用于 Windows、Mac 或 Linux 环境的工具。

2.3. 理解角色和认证

默认情况下，当首次安装时，OpenShift Container Platform 中不会创建角色或用户帐户，因此您需要创建它们。您可以选择创建新角色，也可以定义一个策略来允许任何人登录（以此为基础开始设置）。

在您执行其他操作之前，请使用默认 **system:admin** 用户至少登录一次。在 master 上运行以下命令：

```
$ oc login -u system:admin
```



注意

除非另有说明，否则所有来自现在的命令都应在 主控机 (master) 上执行。

至少使用这个帐户登录一次，您将创建 **system:admin** 用户的配置文件，这将允许您在以后登录。

这个系统帐户没有密码。

运行以下命令验证 OpenShift Container Platform 是否已安装并成功启动。您将获得 master 和节点列表，它们的状态为 **Ready**。

■

```
$ oc get nodes
```

要继续配置您的基本 OpenShift Container Platform 环境，请按照 配置 OpenShift Container Platform 中的步骤进行。

第 3 章 配置 OPENSHIFT CONTAINER PLATFORM

3.1. 概述

本指南将介绍 OpenShift Container Platform 的基本概念，并帮助您配置基本应用程序。本指南提供 安装基本 OpenShift Container Platform 环境后的配置步骤，它不适用于在生产环境中部署或安装 OpenShift。

3.2. 更改身份提供者中的日志

全新安装的 OpenShift Container Platform 实例的默认行为是拒绝任何用户登录。将验证方法改为 HTPasswd:

1. 以编辑模式打开 */etc/origin/master/master-config.yaml* 文件。
2. 查找 **identityProviders** 部分。
3. 将 **DenyAllPasswordIdentityProvider** 改为 **HTPasswdPasswordIdentityProvider**。
4. 将名称标识（label）改为 **htpasswd_auth**，并在 provider 部分添加一个新行 **file: /etc/origin/master/htpasswd**。
   一个带有 **HTPasswdPasswordIdentityProvider** 的 **identityProviders** 项的示例应该和以下类似。

```
oauthConfig:
  ...
  identityProviders:
  - challenge: true
    login: true
    name: htpasswd_auth provider
    provider:
      apiVersion: v1
      kind: HTPasswdPasswordIdentityProvider
      file: /etc/origin/master/htpasswd
```

5. 保存该文件。

3.3. 创建用户帐户

现在，您已开始使用 **HTPasswdPasswordIdentityProvider** 身份验证机制，您需要生成相应的用户账户。

1. 您可以使用 **httpd-tools** 软件包来获得可生成这些帐户的 **htpasswd** 二进制文件。

```
# yum -y install httpd-tools
```

2. 创建用户帐户。

```
# touch /etc/origin/master/htpasswd
# htpasswd -b /etc/origin/master/htpasswd admin redhat
```

   您已创建了用户 **admin**，密码为 **redhat**。

3. 在继续前重启 OpenShift。

```
# master-restart api
# master-restart controllers
```

4. 授予这个用户帐户 **cluster-admin** 权限，这样它就可以完成所有操作。

```
$ oc adm policy add-cluster-role-to-user cluster-admin admin
```

运行 **oc adm** 命令时，您应该只从 Ansible 主机清单文件中列出的第一个 master 运行它们，默认为 */etc/ansible/hosts*。

5. 您可以使用这个用户名/密码组合来通过 web 控制台或命令行登录。为了测试您可以登陆，请运行以下命令：

```
$ oc login -u admin
```

在继续进行前，切换到 **default** 项目。

```
$ oc project default
```

如需更多详细信息，请参阅 角色 和 身份验证。

3.4. 部署 OPENSHIFT 路由器

OpenShift 路由器是用于 OpenShift 服务的外部网络流量的入口点。它支持 HTTP、HTTPS 以及任何启用了 SNI 的 TLS 流量，这使路由器能够向正确的服务发送流量。

没有路由器，OpenShift 服务和 pod 无法与 OpenShift 实例外的任何资源通信。

安装程序会创建一个默认路由器。

1. 使用以下命令删除默认路由器。

```
$ oc delete all -l router=router
```

2. 创建一个新的默认路由器。

```
$ oc adm router --replicas=1 --service-account=router
```

OpenShift 文档包含有关 路由器概述的详细信息。

3.5. 部署内部 REGISTRY

OpenShift 提供了一个内部 集成的容器镜像 registry，可以将其部署来在本地管理镜像。OpenShift 使用 **docker-registry** 来存储、检索和构建容器镜像，并在整个生命周期中部署和管理它们。

安装程序会创建一个默认 registry。

第 4 章 使用 WEB 控制台创建并构建镜像

4.1. 概述

此节介绍了如何通过最简单的方法获取示例项目并在 OpenShift Container Platform 上运行的信息。在一个项目中创建镜像有几种方法，但本主题着重阐述最快捷、最简单的方法。

如果这是您阅读的文档的第一部分，并且您对 OpenShift Container Platform 版本 3 (v3) 的核心概念不熟悉，您可能需要先阅读 新内容。此版本的 OpenShift Container Platform 与版本 2 (v2) 有很大不同。

OpenShift Container Platform 3 提供了一组 编程语言 和 数据库，以及相关的信息和教程，供开发人员参考来快速开始进行应用程序开发。Quickstart 模板提供了编程语言支持，它利用 构建器（builder）镜像。

语言	实施及指南
Ruby	Rails
Python	Django
Node.js	Node.js
PHP	CakePHP
Perl	Dancer
Java	

OpenShift Container Platform 提供的其他镜像包括：

- MySQL
- MongoDB
- PostgreSQL
- Jenkins

另外，JBoss Middleware 还整合了大量 OpenShift Container Platform 模板。

XPaaS 服务提供的技术包括：

- JBoss EAP 6 提供的 Java EE 6 Application Server
- 由 JBoss Fuse 和 JBoss A-MQ 提供的集成和消息服务
- 由 JBoss Data Grid 提供的数据网格服务
- JBoss BRMS 提供的实时决定服务
- Tomcat 7 和 Tomcat 8 提供的 Java Web Server 3.0

用户可以使用这些服务的组合：

- 只使用 HTTP，或使用 HTTP 和 HTTPS
- 不需要数据库，或使用 MongoDB、PostgreSQL 或 MySQL。
- 如果需要，可以与 A-MQ 集成

为了帮助演示这些应用程序，以下小节引导您创建一个包含示例 Node.js 应用程序的项目，该应用程序将提供欢迎页面和当前点击计数（存储在数据库中）。

注意

本节讨论 Quickstart 和 Instant App 模板及应用程序。Quickstarts 为应用程序开发提供了一个起点，您可以以它为基础来创建自己的应用程序。而类似 Jenkins 的 Instant Apps 可立即使用。

4.1.1. 浏览器要求

查看可用于访问 Web 控制台的浏览器版本和操作系统。

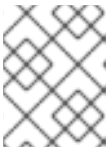
4.2. 开始前

开始前：

- 您需要可以访问运行的 OpenShift Container Platform 4 实例。如果没有访问权限，请联络您的集群管理员。
- 您的实例必须由集群管理员预先配置，并包含 Instant App 模板 和 构建器镜像。如果没有这些内容，请推荐集群管理员参阅 Loading the Default Image Streams and Templates 中的内容。
- 您需要已下载并安装 OpenShift Container Platform CLI 。

4.3. FORK SAMPLE 仓库

1. 登录到 GitHub，访问 Ruby 示例 页面。



注意

本节介绍 Ruby 示例，但您可以使用 OpenShift Container Platform GitHub 项目中提供的任何语言示例进行操作。

2. Fork 仓库。
   您会被重新指向您的新 fork。
3. 复制 fork 的克隆 URL。
4. 将存储库克隆到您的本地机器。

4.4. 创建一个项目

要创建应用程序，您必须首先创建一个新项目，然后选择 InstantApp 模板。从那里，OpenShift Container Platform 开始构建过程并创建一个新的部署。

1. 使用您的网络浏览器访问 OpenShift Container Platform Web 控制台。Web 控制台使用自签名证书，因此如果出现提示，请继续操作。
2. 使用您的系统管理员为您分配的用户名和密码登录。
3. 要创建新项目，请点击 **New Project**。
4. 为新项目输入唯一名称、显示名称和描述。
5. 点击 **Create**。
   Web 控制台的欢迎页面被加载。

4.5. 从镜像创建应用程序

通过 Select Image or Template 页面，您可以选择通过 git 仓库或使用模板进行创建：

1. 如果创建新项目没有自动重定向到 Select Image or Template 页面，您可能需要点击 **Add to Project**。
2. 点击 **Browse**，然后从下拉列表中选择 **ruby**。
3. 点击 **ruby:latest** builder 镜像。
4. 为您的应用程序输入一个 **名称**，并指定 **Git Repository URL**，即 **https://github.com/<your_github_username>/ruby-ex.git**。
5. 在默认情况下，此示例应用程序会自动创建路由、Webhook 触发器和构建更改触发器。您也可以选择使用 **Show advanced routing, build, and deployment options**。
6. 点击 **Create**。

注意

创建后，您可以通过点 **Browse**, **Builds**, 选择您的构建，再点 **Actions**，使用 **Edit** 或 **Edit YAML** 来对其中的一些设置进行修改。

创建应用程序可能需要一些时间。您可以监控 web 控制台的 Overview 页面，以查看创建的新资源，并观察构建和部署的进度。

在创建 Ruby pod 时，它的状态会显示为 pending。然后，Ruby pod 启动并显示其新分配的 IP 地址。当 Ruby pod 运行时，则代表构建已完成。

4.6. 确定应用程序正在运行

如果正确配置了 DNS，那么就可以使用 Web 浏览器访问您的新应用程序。如果无法访问您的应用程序，请联系系统管理员。

查看您的新应用程序：

4.7. 配置自动构建

从 OpenShift Container Platform GitHub 仓库 获取此应用程序的源代码。因此，每当您将代码更改推送到您的分叉仓库时，webhook 会自动触发应用程序的重新构建过程。

为您的应用程序设置 webhook:

1. 从 Web 控制台导航至包含应用程序的项目。
2. 点 **Browse** 选项卡，然后点 **Builds**。
3. 点您的构建名称，然后点击 **Configuration** 选项卡。
4. 点击  接下来到 **GitHub Webhook URL** 以复制 webhook URL。
5. 进入 GitHub 上 fork 的仓库，然后点 **Settings**。
6. 点击 **Webhooks & Services**。
7. 点击 **Add webhook**。
8. 将 webhook URL 粘贴到 **Payload URL** 字段。
9. 将 **Content Type** 设置为 **application/json**。
10. 点击 **Add webhook**。

GitHub 现在会尝试向 OpenShift Container Platform 服务器发送 ping 有效负载，以确保通信成功。如果您在 Webhook URL 旁边看到一个绿色检查标记，则代表它被正确配置。鼠标悬停在检查标记之上，可查看最后一次发送的状态。

当您下一次将代码更改推送到 fork 仓库时，应用程序会自动重建。

4.8. 编写代码更改

要在本地工作，然后将更改推送到应用程序：

1. 在您的本地机器上，使用文本编辑器来更改文件 ***ruby-ex/config.ru***的示例应用程序源代码。
2. 在您的应用程序中进行代码更改是可见的。例如： 在第 229 行中，将标题由 **Welcome to your Ruby application on OpenShift** 改为 **This is my Awesome OpenShift Application**，然后保存改变。
3. 使用 git 提交更改，并将更改推送至您的 fork。
   如果您正确配置了 webhook，则应用程序会立刻根据您的更改重新构建其自身。重建成功后，使用之前创建的路由查看更新的应用程序。

现在，所有您需要做的工作都是推送代码更新，OpenShift Container Platform 会处理剩余的操作。

4.8.1. 手动重建镜像

如果 webhook 无法正常工作，或者构建失败，且您不想在重启构建前更改代码， 则可以手动重建镜像。使用以下方法根据您最新的代码手动重建镜像：

1. 点 **Browse** 选项卡，然后点 **Builds**。
2. 找到构建，然后点击 **Start Build**。

第 5 章 使用 CLI 创建并构建镜像

5.1. 概述

此节介绍了如何通过最简单的方法获取示例项目并在 OpenShift Container Platform 上运行的信息。在一个项目中创建镜像有几种方法，但本主题着重阐述最快捷、最简单的方法。

如果这是您阅读的文档的第一部分，并且您对 OpenShift Container Platform 版本 3 (v3) 的核心概念不熟悉，您可能需要先阅读 新内容。此版本的 OpenShift Container Platform 与版本 2 (v2) 有很大不同。

OpenShift Container Platform 3 提供了一组 编程语言 和 数据库，以及相关的信息和教程，供开发人员参考来快速开始进行应用程序开发。Quickstart 模板提供了编程语言支持，它利用 构建器（builder）镜像。

语言	实施及指南
Ruby	Rails
Python	Django
Node.js	Node.js
PHP	CakePHP
Perl	Dancer
Java	

OpenShift Container Platform 提供的其他镜像包括：

- MySQL
- MongoDB
- PostgreSQL
- Jenkins

另外，JBoss Middleware 还整合了大量 OpenShift Container Platform 模板。

XPaaS 服务提供的技术包括：

- JBoss EAP 6 提供的 Java EE 6 Application Server
- 由 JBoss Fuse 和 JBoss A-MQ 提供的集成和消息服务
- 由 JBoss Data Grid 提供的数据网格服务
- JBoss BRMS 提供的实时决定服务
- Tomcat 7 和 Tomcat 8 提供的 Java Web Server 3.0

用户可以使用这些服务的组合：

- 只使用 HTTP，或使用 HTTP 和 HTTPS
- 不需要数据库，或使用 MongoDB、PostgreSQL 或 MySQL。
- 如果需要，可以与 A-MQ 集成

为了帮助演示这些应用程序，以下小节引导您创建一个包含示例 Node.js 应用程序的项目，该应用程序将提供欢迎页面和当前点击计数（存储在数据库中）。

注意

本节讨论 Quickstart 和 Instant App 模板及应用程序。Quickstarts 为应用程序开发提供了一个起点，您可以以它为基础来创建自己的应用程序。而类似 Jenkins 的 Instant Apps 可立即使用。

5.2. 开始前

开始前：

- 您需要可以访问运行的 OpenShift Container Platform 4 实例。如果没有访问权限，请联络您的集群管理员。
- 您的实例必须由集群管理员预先配置，并包含 Instant App 模板 和 构建器镜像。如果没有这些内容，请推荐集群管理员参阅 Loading the Default Image Streams and Templates 中的内容。
- 您需要已下载并安装 OpenShift Container Platform CLI 。

5.3. FORK SAMPLE 仓库

1. 登录到 GitHub，访问 Ruby 示例 页面。

注意

本节介绍 Ruby 示例，但您可以使用 OpenShift Container Platform GitHub 项目中提供的任何语言示例进行操作。

2. Fork 仓库。
   您会被重新指向您的新 fork。
3. 复制 fork 的克隆 URL。
4. 将存储库克隆到您的本地机器。

5.4. 创建一个项目

要创建应用程序，您必须创建一个新项目并指定源的位置。从那里，OpenShift Container Platform 开始构建过程并创建一个新的部署。

1. 通过 CLI 登陆到 OpenShift Container Platform：
   - 使用用户名和密码登陆：

```
$ oc login -u=<username> -p=<password> --server=<your-openshift-server> --insecure-skip-tls-verify
```

- 使用 oauth 令牌登陆：

```
$ oc login <https://api.your-openshift-server.com> --token=<tokenID>
```

2. 创建一个新项目

```
$ oc new-project <projectname> --description="<description>" --display-name="<display_name>"
```

在创建了新项目后，您将自动切换到新项目的命名空间。

5.5. 从镜像创建应用程序

使用您 fork 的仓库中的代码创建新应用程序：

1. 通过指定代码的源来创建应用程序：

```
$ oc new-app openshift/ruby-20-centos7~https://github.com/<your_github_username>/ruby-ex
```

OpenShift Container Platform 找到匹配的构建器镜像（本例中为 **ruby-20-centos7**），然后为应用程序创建资源（镜像流、构建配置、部署配置和服务）。它还会调度构建。

2. 监控构建的进度：

```
$ oc logs -f bc/ruby-ex
```

3. 构建完成后，生成的镜像成功推送到 registry，检查应用程序的状态：

```
$ oc status
```

或者，您可以从 Web 控制台查看构建。

创建应用程序可能需要一些时间。您可以监控 web 控制台的 Overview 页面，以查看创建的新资源，并观察构建和部署的进度。您还可以使用 **oc get pods** 命令检查 pod 的启动和运行时间，或 **oc get builds** 命令来查看构建统计信息。

在创建 Ruby pod 时，它的状态会显示为 pending。然后，Ruby pod 启动并显示其新分配的 IP 地址。当 Ruby pod 运行时，则代表构建已完成。

oc status 命令告诉您服务正在运行的 IP 地址，其部署的默认端口为 8080。

5.6. 创建路由

OpenShift Container Platform 路由以主机名的形式公开服务，以便外部客户端可根据名称访问该服务。创建指向新应用程序的路由：

1. 为 **ruby-ex** 开放服务：

```
$ oc expose service ruby-ex
```

2. 查看您的新路由：

```
$ oc get route
```

5.7. 确定应用程序正在运行

要查看您的新应用程序，请路由位置（在上一步中获得）复制到网页浏览器的地址栏里，然后按 Enter 键。

ruby-ex 应用程序示例带有一个简单的欢迎页面，它包括了如何部署代码更改、管理应用程序和其他开发资源的详情。

接下来，通过 GitHub webhook 触发器配置自动构建，从而可以当 fork 仓库中的代码发生变化时自动重新构建应用程序。

5.8. 配置自动构建

从 OpenShift Container Platform GitHub 仓库获取此应用程序的源代码。因此，每当您将代码更改推送到您的分叉仓库时，webhook 会自动触发应用程序的重新构建过程。

为您的应用程序设置 webhook:

1. 查看 **BuildConfig** 中的 triggers 部分来检查 GitHub webhook trigger 是否存在：

```
$ oc edit bc/ruby-ex
```

   您应该可以看到类似如下的内容：

```
triggers
- github:
    secret: Q1tGY0i9f1ZFihQbX07S
    type: GitHub
```

   secret 可确保只有您和您的仓库可以触发构建。

2. 您可以使用以下命令来显示与 **BuildConfig** 关联的 webhook URL。

```
$ oc describe bc ruby-ex
```

3. 按以上命令复制 GitHub webhook 有效负载 URL 输出。
4. 进入 GitHub 上 fork 的仓库，然后点 **Settings**。
5. 点击 **Webhooks & Services**。
6. 点击 **Add webhook**。
7. 将 webhook URL 粘贴到 **Payload URL** 字段。
8. 将 **Content Type** 设置为 **application/json**。
9. 点击 **Add webhook**。

GitHub 现在会尝试向 OpenShift Container Platform 服务器发送 ping 有效负载，以确保通信成功。如果您在 Webhook URL 旁边看到一个绿色检查标记，则代表它被正确配置。鼠标悬停在检查标记之上，可查看最后一次发送的状态。

当您下一次将代码更改推送到 fork 仓库时，应用程序会自动重建。

5.9. 编写代码更改

要在本地工作，然后将更改推送到应用程序：

1. 在您的本地机器上，使用文本编辑器来更改文件 ***ruby-ex/config.ru***的示例应用程序源代码。
2. 在您的应用程序中进行代码更改是可见的。例如： 在第 229 行中，将标题由 **Welcome to your Ruby application on OpenShift** 改为 **This is my Awesome OpenShift Application**，然后保存改变。
3. 使用 git 提交更改，并将更改推送至您的 fork。
   如果您正确配置了 webhook，则应用程序会立刻根据您的更改重新构建其自身。重建成功后，使用之前创建的路由查看更新的应用程序。

现在，所有您需要做的工作都是推送代码更新，OpenShift Container Platform 会处理剩余的操作。

5.9.1. 手动重建镜像

如果 webhook 无法正常工作，或者构建失败，且您不想在重启构建前更改代码， 则可以手动重建镜像。使用以下方法根据您最新的代码手动重建镜像：

```
$ oc start-build ruby-ex
```

5.10. 故障排除

改变项目

虽然 **oc new-project** 命令会自动将当前项目设置为您刚刚创建的项目，但您始终可以通过运行以下命令来更改项目：

```
$ oc project <project-name>
```

要查看项目列表，请运行：

```
$ oc get projects
```

手动触发构建

如果构建没有自动启动，则启动构建并输出日志：

```
$ oc start-build ruby-ex --follow
```

或者，不要在上述命令中包含 **--follow**，而是在触发构建后使用以下命令，其中 **n** 是要跟踪的构建号：

```
$ oc logs -f build/ruby-ex-n
```