



OpenShift Container Platform 4.7

Web console

Getting started with the web console in OpenShift Container Platform

OpenShift Container Platform 4.7 Web console

Getting started with the web console in OpenShift Container Platform

Legal Notice

Copyright © 2022 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document provides instructions for accessing and customizing the OpenShift Container Platform web console.

Table of Contents

CHAPTER 1. WEB CONSOLE OVERVIEW	4
1.1. ABOUT THE ADMINISTRATOR PERSPECTIVE IN THE WEB CONSOLE	4
1.1.1. Accessing the Administrator perspective	4
1.2. ABOUT THE DEVELOPER PERSPECTIVE IN THE WEB CONSOLE	5
1.2.1. Accessing the Developer perspective	5
CHAPTER 2. ACCESSING THE WEB CONSOLE	7
2.1. PREREQUISITES	7
2.2. UNDERSTANDING AND ACCESSING THE WEB CONSOLE	7
CHAPTER 3. USING THE OPENSIFT CONTAINER PLATFORM DASHBOARD TO GET CLUSTER INFORMATION	8
3.1. ABOUT THE OPENSIFT CONTAINER PLATFORM DASHBOARDS PAGE	8
CHAPTER 4. CONFIGURING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM	10
4.1. PREREQUISITES	10
4.2. CONFIGURING THE WEB CONSOLE	10
CHAPTER 5. CUSTOMIZING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM	11
5.1. ADDING A CUSTOM LOGO AND PRODUCT NAME	11
5.2. CREATING CUSTOM LINKS IN THE WEB CONSOLE	12
5.3. CUSTOMIZING THE WEB CONSOLE URL	13
5.4. CUSTOMIZING THE LOGIN PAGE	14
5.5. DEFINING A TEMPLATE FOR AN EXTERNAL LOG LINK	15
5.6. CREATING CUSTOM NOTIFICATION BANNERS	16
5.7. CUSTOMIZING CLI DOWNLOADS	16
5.8. ADDING YAML EXAMPLES TO KUBERNETES RESOURCES	17
CHAPTER 6. ABOUT THE WEB TERMINAL IN THE WEB CONSOLE	19
6.1. INSTALLING THE WEB TERMINAL	19
6.2. USING THE WEB TERMINAL	20
6.3. UNINSTALLING THE WEB TERMINAL	20
6.3.1. Deleting the web terminal components and custom resources	21
6.3.2. Uninstalling the Operator using the web console	22
CHAPTER 7. DISABLING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM	23
7.1. PREREQUISITES	23
7.2. DISABLING THE WEB CONSOLE	23
CHAPTER 8. CREATING QUICK START TUTORIALS IN THE WEB CONSOLE	24
8.1. UNDERSTANDING QUICK STARTS	24
8.2. QUICK START USER WORKFLOW	24
8.3. QUICK START COMPONENTS	25
8.4. CONTRIBUTING QUICK STARTS	25
8.4.1. Viewing the quick start API documentation	26
8.4.2. Mapping the elements in the quick start to the quick start CR	26
8.4.2.1. conclusion element	26
8.4.2.2. description element	27
8.4.2.3. displayName element	28
8.4.2.4. durationMinutes element	29
8.4.2.5. icon element	30
8.4.2.6. introduction element	32
8.4.3. Adding a custom icon to a quick start	34

8.4.4. Limiting access to a quick start	35
8.4.5. Linking to other quick starts	35
8.4.6. Supported tags for quick starts	35
8.4.7. Quick start highlighting markdown reference	36
8.4.7.1. Perspective switcher	37
8.4.7.2. Administrator perspective navigation links	37
8.4.7.3. Developer perspective navigation links	37
8.4.7.4. Common navigation links	37
8.4.7.5. Masthead links	37
8.5. QUICK START CONTENT GUIDELINES	37
8.5.1. Card copy	37
8.5.2. Introduction	38
8.5.3. Task steps	38
8.5.4. Check your work module	40
8.5.5. Formatting UI elements	40
8.6. ADDITIONAL RESOURCES	41

CHAPTER 1. WEB CONSOLE OVERVIEW

The Red Hat OpenShift Container Platform web console provides a graphical user interface to visualize your project data and perform administrative, management, and troubleshooting tasks. The web console runs as pods on the control plane nodes in the openshift-console project. It is managed by a **console-operator** pod. Both **Administrator** and **Developer** perspectives are supported.

Both **Administrator** and **Developer** perspectives enable you to create quick start tutorials for OpenShift Container Platform. A quick start is a guided tutorial with user tasks and is useful for getting oriented with an application, Operator, or other product offering.

1.1. ABOUT THE ADMINISTRATOR PERSPECTIVE IN THE WEB CONSOLE

The **Administrator** perspective enables you to view the cluster inventory, capacity, general and specific utilization information, and the stream of important events, all of which help you to simplify planning and troubleshooting tasks. Both project administrators and cluster administrators can view the **Administrator** perspective.

Cluster administrators can also open an embedded command line terminal instance with the web terminal Operator in OpenShift Container Platform 4.7 and later.



NOTE

The default web console perspective that is shown depends on the role of the user. The **Administrator** perspective is displayed by default if the user is recognized as an administrator.

The **Administrator** perspective provides workflows specific to administrator use cases, such as the ability to:

- Manage workload, storage, networking, and cluster settings.
- Install and manage Operators using the Operator Hub.
- Add identity providers that allow users to log in and manage user access through roles and role bindings.
- View and manage a variety of advanced settings such as cluster updates, partial cluster updates, cluster Operators, custom resource definitions (CRDs), role bindings, and resource quotas.
- Access and manage monitoring features such as metrics, alerts, and monitoring dashboards.
- View and manage logging, metrics, and high-status information about the cluster.
- Visually interact with applications, components, and services associated with the **Administrator** perspective in OpenShift Container Platform.

Additional resources

See [About the web terminal in the web console](#) for more information on the web terminal Operator.

1.1.1. Accessing the Administrator perspective

The **Administrator** perspective in the OpenShift Container Platform web console provides workflows specific to administrator use cases. You can access the **Administrator** perspective from the web console as follows:

Procedure

- Log in to the OpenShift Container Platform web console using your login credentials to access the **Administrator** perspective.

1.2. ABOUT THE DEVELOPER PERSPECTIVE IN THE WEB CONSOLE

The **Developer** perspective offers several built-in ways to deploy applications, services, and databases. In the **Developer** perspective, you can:

- View real-time visualization of rolling and recreating rollouts on the component.
- View the application status, resource utilization, project event streaming, and quota consumption.
- Share your project with others.
- Troubleshoot problems with your applications by running Prometheus Query Language (PromQL) queries on your project and examining the metrics visualized on a plot. The metrics provide information about the state of a cluster and any user-defined workloads that you are monitoring.

Cluster administrators can also open an embedded command line terminal instance in the web console in OpenShift Container Platform 4.7 and later.



NOTE

The default web console perspective that is shown depends on the role of the user. The **Developer** perspective is displayed by default if the user is recognised as a developer.

The **Developer** perspective provides workflows specific to developer use cases, such as the ability to:

- Create and deploy applications on OpenShift Container Platform by importing existing codebases, images, and container files.
- Visually interact with applications, components, and services associated with them within a project and monitor their deployment and build status.
- Group components within an application and connect the components within and across applications.
- Integrate serverless capabilities (Technology Preview).
- Create workspaces to edit your application code using Eclipse Che.

1.2.1. Accessing the Developer perspective

The **Developer** perspective in the OpenShift Container Platform web console provides workflows specific to developer use cases.

You can access the **Developer** perspective from the web console as follows:

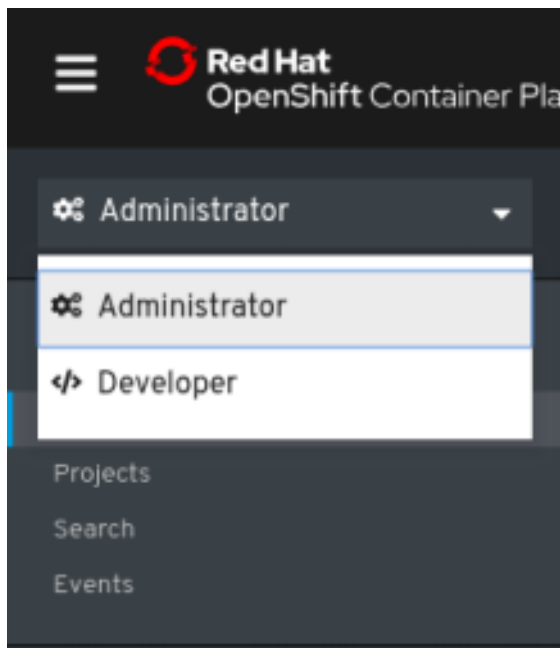
Prerequisites

To access the **Developer** perspective, ensure that you have logged in to the web console. The default view for the OpenShift Container Platform web console is the **Developer** perspective.

Procedure

1. Use the perspective switcher to switch to the **Developer** perspective. The **Topology** view with a list of all the projects in your cluster is displayed.

Figure 1.1. Developer perspective



2. Select an existing project from the list or use the **Project** drop-down list to create a new project.

If you have no workloads or applications in the project, the **Topology** view displays the available options to create applications. If you have existing workloads, the **Topology** view graphically displays your workload nodes.

Additional resources

- [Learn more about Cluster Administrator](#)
- [Viewing cluster information](#)
- [Configuring the web console](#)
- [Customizing the web console](#)
- [Launching an embedded command line terminal instance in the web console](#)
- [Creating quick start tutorials](#)
- [Disabling the web console](#)

CHAPTER 2. ACCESSING THE WEB CONSOLE

The OpenShift Container Platform web console is a user interface accessible from a web browser. Developers can use the web console to visualize, browse, and manage the contents of projects.

2.1. PREREQUISITES

- JavaScript must be enabled to use the web console. For the best experience, use a web browser that supports [WebSockets](#).
- Review the [OpenShift Container Platform 4.x Tested Integrations](#) page before you create the supporting infrastructure for your cluster.

2.2. UNDERSTANDING AND ACCESSING THE WEB CONSOLE

The web console runs as a pod on the master. The static assets required to run the web console are served by the pod. Once OpenShift Container Platform is successfully installed using **openshift-install create cluster**, find the URL for the web console and login credentials for your installed cluster in the CLI output of the installation program. For example:

Example output

```
INFO Install complete!
INFO Run 'export KUBECONFIG=<your working directory>/auth/kubeconfig' to manage the cluster
with 'oc', the OpenShift CLI.
INFO The cluster is ready when 'oc login -u kubeadmin -p <provided>' succeeds (wait a few minutes).
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.demo1.openshift4-beta-abcorp.com
INFO Login to the console with user: kubeadmin, password: <provided>
```

Use those details to log in and access the web console.

For existing clusters that you did not install, you can use **oc whoami --show-console** to see the web console URL.

CHAPTER 3. USING THE OPENSIFT CONTAINER PLATFORM DASHBOARD TO GET CLUSTER INFORMATION

Access the OpenShift Container Platform dashboard, which captures high-level information about the cluster, by navigating to **Home** → **Dashboards** → **Overview** from the OpenShift Container Platform web console.

The OpenShift Container Platform dashboard provides various cluster information, captured in individual dashboard cards.

3.1. ABOUT THE OPENSIFT CONTAINER PLATFORM DASHBOARDS PAGE

The OpenShift Container Platform dashboard consists of the following cards:

- **Details** provides a brief overview of informational cluster details. Status include **ok**, **error**, **warning**, **in progress**, and **unknown**. Resources can add custom status names.
 - Cluster ID
 - Provider
 - Version
- **Cluster Inventory** details number of resources and associated statuses. It is helpful when intervention is required to resolve problems, including information about:
 - Number of nodes
 - Number of pods
 - Persistent storage volume claims
 - Bare metal hosts in the cluster, listed according to their state (only available in **metal3** environment).
- **Cluster Capacity** charts help administrators understand when additional resources are required in the cluster. The charts contain an inner ring that displays current consumption, while an outer ring displays thresholds configured for the resource, including information about:
 - CPU time
 - Memory allocation
 - Storage consumed
 - Network resources consumed
- **Cluster Utilization** shows the capacity of various resources over a specified period of time, to help administrators understand the scale and frequency of high resource consumption.
- **Events** lists messages related to recent activity in the cluster, such as pod creation or virtual machine migration to another host.

- **Top Consumers** helps administrators understand how cluster resources are consumed. Click on a resource to jump to a detailed page listing pods and nodes that consume the largest amount of the specified cluster resource (CPU, memory, or storage).

CHAPTER 4. CONFIGURING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM

You can modify the OpenShift Container Platform web console to set a logout redirect URL or disable the console.

4.1. PREREQUISITES

- Deploy an OpenShift Container Platform cluster.

4.2. CONFIGURING THE WEB CONSOLE

You can configure the web console settings by editing the **console.config.openshift.io** resource.

- Edit the **console.config.openshift.io** resource:

```
$ oc edit console.config.openshift.io cluster
```

The following example displays the sample resource definition for the console:

```
apiVersion: config.openshift.io/v1
kind: Console
metadata:
  name: cluster
spec:
  authentication:
    logoutRedirect: "" 1
status:
  consoleURL: "" 2
```

- 1 Specify the URL of the page to load when a user logs out of the web console. If you do not specify a value, the user returns to the login page for the web console. Specifying a **logoutRedirect** URL allows your users to perform single logout (SLO) through the identity provider to destroy their single sign-on session.
- 2 The web console URL. To update this to a custom value, see **Customizing the web console URL**.

CHAPTER 5. CUSTOMIZING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM

You can customize the OpenShift Container Platform web console to set a custom logo, product name, links, notifications, and command line downloads. This is especially helpful if you need to tailor the web console to meet specific corporate or government requirements.

5.1. ADDING A CUSTOM LOGO AND PRODUCT NAME

You can create custom branding by adding a custom logo or custom product name. You can set both or one without the other, as these settings are independent of each other.

Prerequisites

- You must have administrator privileges.
- Create a file of the logo that you want to use. The logo can be a file in any common image format, including GIF, JPG, PNG, or SVG, and is constrained to a **max-height** of **60px**.

Procedure

1. Import your logo file into a config map in the **openshift-config** namespace:

```
$ oc create configmap console-custom-logo --from-file /path/to/console-custom-logo.png -n openshift-config
```

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: console-custom-logo
  namespace: openshift-config
data:
  console-custom-logo.png: <base64-encoded_logo> ... 1
```

- 1 Provide a valid base64 encoded logo.

2. Edit the web console's Operator configuration to include **customLogoFile** and **customProductName**:

```
$ oc edit consoles.operator.openshift.io cluster
```

```
apiVersion: operator.openshift.io/v1
kind: Console
metadata:
  name: cluster
spec:
  customization:
    customLogoFile:
      key: console-custom-logo.png
      name: console-custom-logo
    customProductName: My Console
```

Once the Operator configuration is updated, it will sync the custom logo config map into the console namespace, mount it to the console pod, and redeploy.

3. Check for success. If there are any issues, the console cluster Operator will report a **Degraded** status, and the console Operator configuration will also report a **CustomLogoDegraded** status, but with reasons like **KeyOrFilenameInvalid** or **NoImageProvided**.

To check the **clusteroperator**, run:

```
$ oc get clusteroperator console -o yaml
```

To check the console Operator configuration, run:

```
$ oc get consoles.operator.openshift.io -o yaml
```

5.2. CREATING CUSTOM LINKS IN THE WEB CONSOLE

Prerequisites

- You must have administrator privileges.

Procedure

1. From **Administration** → **Custom Resource Definitions** click on **ConsoleLink**.
2. Select **Instances** tab
3. Click **Create Console Link** and edit the file:

```
apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: example
spec:
  href: 'https://www.example.com'
  location: HelpMenu 1
  text: Link 1
```

- 1** Valid location settings are **HelpMenu**, **UserMenu**, **ApplicationMenu**, and **NamespaceDashboard**.

To make the custom link appear in all namespaces, follow this example:

```
apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: namespaced-dashboard-link-for-all-namespaces
spec:
  href: 'https://www.example.com'
  location: NamespaceDashboard
  text: This appears in all namespaces
```

To make the custom link appear in only some namespaces, follow this example:

-


```

apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: namespaced-dashboard-for-some-namespaces
spec:
  href: 'https://www.example.com'
  location: NamespaceDashboard
  # This text will appear in a box called "Launcher" under "namespace" or "project" in the web
  console
  text: Custom Link Text
  namespaceDashboard:
    namespaces:
      # for these specific namespaces
      - my-namespace
      - your-namespace
      - other-namespace

```

To make the custom link appear in the application menu, follow this example:

```

apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: application-menu-link-1
spec:
  href: 'https://www.example.com'
  location: ApplicationMenu
  text: Link 1
  applicationMenu:
    section: My New Section
    # image that is 24x24 in size
    imageURL: https://via.placeholder.com/24

```

4. Click **Save** to apply your changes.

5.3. CUSTOMIZING THE WEB CONSOLE URL

You can update the web console URL, **consoleURL**, to a custom value.

Procedure

1. Modify the cluster instance created by default during installation in the **consoles.operator.openshift.io** custom resource:

```

$ oc patch consoles.operator.openshift.io cluster --patch '{"spec":{"route":{"hostname":"console.example.com"}}}' --type=merge

```

```

apiVersion: operator.openshift.io/v1
kind: Console
metadata:
  name: cluster
spec:
  route:
    hostname: 'console.example.com'

```

- If you specify a custom certificate, you must create a secret in the **openshift-config** namespace that has the key and certificate. For example:

```
$ oc create secret tls console-tls --key=key.pem --cert=cert.pem -n openshift-config
```

```
apiVersion: v1
kind: Secret
metadata:
  name: console-tls
  namespace: openshift-config
type: kubernetes.io/tls
data:
  tls.crt: >-
    <base64-encoded_cert> ... 1
  tls.key: >-
    <base64-encoded_key> ... 2
```

- Provide a valid TLS certificate.
- Provide a valid TLS key.

Then, edit the web console's Operator configuration:

```
$ oc edit consoles.operator.openshift.io cluster
```

Add the following stanza to the configuration resource:

```
spec:
  route:
    hostname: console.example.com
  secret:
    name: console-tls
```

5.4. CUSTOMIZING THE LOGIN PAGE

Create Terms of Service information with custom login pages. Custom login pages can also be helpful if you use a third-party login provider, such as GitHub or Google, to show users a branded page that they trust and expect before being redirected to the authentication provider. You can also render custom error pages during the authentication process.



NOTE

Customizing the error template is limited to identity providers (IDPs) that use redirects, such as request header and OIDC-based IDPs. It does not have an effect on IDPs that use direct password authentication, such as LDAP and HTTPasswd.

Prerequisites

- You must have administrator privileges.

Procedure

1. Run the following commands to create templates you can modify:

```
$ oc adm create-login-template > login.html
```

```
$ oc adm create-provider-selection-template > providers.html
```

```
$ oc adm create-error-template > errors.html
```

2. Create the secrets:

```
$ oc create secret generic login-template --from-file=login.html -n openshift-config
```

```
$ oc create secret generic providers-template --from-file=providers.html -n openshift-config
```

```
$ oc create secret generic error-template --from-file=errors.html -n openshift-config
```

3. Run:

```
$ oc edit oauths cluster
```

4. Update the specification:

```
spec:
  templates:
    error:
      name: error-template
    login:
      name: login-template
    providerSelection:
      name: providers-template
```

Run **oc explain oauths.spec.templates** to understand the options.

5.5. DEFINING A TEMPLATE FOR AN EXTERNAL LOG LINK

If you are connected to a service that helps you browse your logs, but you need to generate URLs in a particular way, then you can define a template for your link.

Prerequisites

- You must have administrator privileges.

Procedure

1. From **Administration** → **Custom Resource Definitions**, click on **ConsoleExternalLogLink**.
2. Select **Instances** tab
3. Click **Create Console External Log Link** and edit the file:

```
apiVersion: console.openshift.io/v1
kind: ConsoleExternalLogLink
```

```

metadata:
  name: example
spec:
  hrefTemplate: >-
    https://example.com/logs?
resourceName=${resourceName}&containerName=${containerName}&resourceNamespace=${
resourceNamespace}&podLabels=${podLabels}
text: Example Logs

```

5.6. CREATING CUSTOM NOTIFICATION BANNERS

Prerequisites

- You must have administrator privileges.

Procedure

- From **Administration** → **Custom Resource Definitions**, click on **ConsoleNotification**.
- Select **Instances** tab
- Click **Create Console Notification** and edit the file:

```

apiVersion: console.openshift.io/v1
kind: ConsoleNotification
metadata:
  name: example
spec:
  text: This is an example notification message with an optional link.
  location: BannerTop 1
  link:
    href: 'https://www.example.com'
    text: Optional link text
  color: '#fff'
  backgroundColor: '#0088ce'

```

- 1** Valid location settings are **BannerTop**, **BannerBottom**, and **BannerTopBottom**.

- Click **Create** to apply your changes.

5.7. CUSTOMIZING CLI DOWNLOADS

You can configure links for downloading the CLI with custom link text and URLs, which can point directly to file packages or to an external page that provides the packages.

Prerequisites

- You must have administrator privileges.

Procedure

- Navigate to **Administration** → **Custom Resource Definitions**

2. Select **ConsoleCLIDownload** from the list of Custom Resource Definitions (CRDs).
3. Click the **YAML** tab, and then make your edits:

```

apiVersion: console.openshift.io/v1
kind: ConsoleCLIDownload
metadata:
  name: example-cli-download-links-for-foo
spec:
  description: |
    This is an example of download links for foo
  displayName: example-foo
  links:
  - href: 'https://www.example.com/public/foo.tar'
    text: foo for linux
  - href: 'https://www.example.com/public/foo.mac.zip'
    text: foo for mac
  - href: 'https://www.example.com/public/foo.win.zip'
    text: foo for windows

```

4. Click the **Save** button.

5.8. ADDING YAML EXAMPLES TO KUBERNETES RESOURCES

You can dynamically add YAML examples to any Kubernetes resources at any time.

Prerequisites

- You must have cluster administrator privileges.

Procedure

1. From **Administration** → **Custom Resource Definitions**, click on **ConsoleYAMLSample**.
2. Click **YAML** and edit the file:

```

apiVersion: console.openshift.io/v1
kind: ConsoleYAMLSample
metadata:
  name: example
spec:
  targetResource:
    apiVersion: batch/v1
    kind: Job
  title: Example Job
  description: An example Job YAML sample
  yaml: |
    apiVersion: batch/v1
    kind: Job
    metadata:
      name: countdown
    spec:
      template:
        metadata:
          name: countdown

```

```
spec:  
  containers:  
  - name: counter  
    image: centos:7  
    command:  
    - "bin/bash"  
    - "-c"  
    - "for i in 9 8 7 6 5 4 3 2 1 ; do echo $i ; done"  
  restartPolicy: Never
```

Use **spec.snippet** to indicate that the YAML sample is not the full YAML resource definition, but a fragment that can be inserted into the existing YAML document at the user's cursor.

3. Click **Save**.

CHAPTER 6. ABOUT THE WEB TERMINAL IN THE WEB CONSOLE

You can launch an embedded command line terminal instance in the OpenShift web console. You must first install the Web Terminal Operator to use the web terminal.



NOTE

Cluster administrators can access the web terminal in OpenShift Container Platform 4.7 and later.

This terminal instance is preinstalled with common CLI tools for interacting with the cluster, such as **oc**, **kubectl**, **odo**, **kn**, **tkn**, **helm**, **kubens**, **subctl**, and **kubectx**. It also has the context of the project you are working on and automatically logs you in using your credentials.



IMPORTANT

Web terminal is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview/>.

6.1. INSTALLING THE WEB TERMINAL

You can install the web terminal using the Web Terminal Operator listed in the OpenShift Container Platform OperatorHub. When you install the Web Terminal Operator, the custom resource definitions (CRDs) that are required for the command line configuration, such as the **DevWorkspace** CRD, are automatically installed. The web console creates the required resources when you open the web terminal.

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** permissions.


Procedure

1. In the **Administrator** perspective of the web console, navigate to **Operators → OperatorHub**.
2. Use the **Filter by keyword** box to search for the **Web Terminal** Operator in the catalog, and then click the **Web Terminal** tile.
3. Read the brief description about the Operator on the **Web Terminal** page, and then click **Install**.
4. On the **Install Operator** page, retain the default values for all fields.
 - The **alpha** option in the **Update Channel** menu enables installation of the latest release of the Web Terminal Operator.

- The **All namespaces on the cluster** option in the **Installation Mode** menu enables the Operator to watch and be available to all namespaces in the cluster.
 - The **openshift-operators** option in the **Installed Namespace** menu installs the Operator in the default **openshift-operators** namespace.
 - The **Automatic** option in the **Approval Strategy** menu ensures that the future upgrades to the Operator are handled automatically by the Operator Lifecycle Manager.
5. Click **Install**.
 6. In the **Installed Operators** page, click the **View operator** to verify that the Operator is listed on the **Installed Operators** page.
 7. After the Operator is installed, refresh your page to see the command line terminal icon on the upper right of the console.

6.2. USING THE WEB TERMINAL

After the Web Terminal Operator is installed, you can use the web terminal as follows:

1. To launch the web terminal, click the command line terminal icon () on the upper right of the console. A web terminal instance is displayed in the **Command line terminal** pane. This instance is automatically logged in with your credentials.
2. Select the project where the **DevWorkspace** CR must be created from the **Project** drop-down list. By default, the current project is selected.



NOTE

- The **DevWorkspace** CR is created only if it does not already exist.
- The **openshift-terminal** project is the default project used for cluster administrators. They do not have the option to choose another project.

3. Click **Start** to initialize the web terminal using the selected project.

After the web terminal is initialized, you can use the preinstalled CLI tools like **oc**, **kubectrl**, **odo**, **kn**, **tkn**, **helm**, **kubens**, **subctl**, and **kubectx** in the web terminal.

6.3. UNINSTALLING THE WEB TERMINAL

Uninstalling the web terminal is a two-step process:

1. Delete the components and custom resources (CRs) that were added when you installed the Operator.
2. Uninstall the Web Terminal Operator.

Uninstalling the Web Terminal Operator does not remove any of its custom resource definitions (CRDs) or managed resources that are created when the Operator is installed. These components must be manually uninstalled for security purposes. Removing these components also allows you to save cluster resources by ensuring that terminals do not idle when the Operator is uninstalled.

Prerequisites

- Access to an OpenShift Container Platform cluster using an account with **cluster-admin** permissions.

6.3.1. Deleting the web terminal components and custom resources

Use the CLI to delete the CRs that are created during installation of the Web Terminal Operator.

Procedure

1. Run the following commands to ensure that all **DevWorkspace** CRs are removed along with their related Kubernetes objects, such as deployments.

```
$ oc delete devworkspaces.workspace.devfile.io --all-namespaces --all --wait
```

```
$ oc delete workspaceroutings.controller.devfile.io --all-namespaces --all --wait
```

```
$ oc delete components.controller.devfile.io --all-namespaces --all --wait
```



WARNING

If this step is not complete, finalizers make it difficult to fully uninstall the Operator easily.

2. Run the following commands to remove the CRDs:

```
$ oc delete customresourcedefinitions.apiextensions.k8s.io  
workspaceroutings.controller.devfile.io
```

```
$ oc delete customresourcedefinitions.apiextensions.k8s.io components.controller.devfile.io
```

```
$ oc delete customresourcedefinitions.apiextensions.k8s.io  
devworkspaces.workspace.devfile.io
```

3. Remove the **DevWorkspace-Webhook-Server** deployment:

```
$ oc delete deployment/devworkspace-webhook-server -n openshift-operators
```



NOTE

When you run this and the following steps, you cannot use the **oc exec** commands to run commands in a container. After you remove the webhooks you will be able to use the **oc exec** commands again.

4. Run the following commands to remove any lingering services, secrets, and config maps:

■

```
$ oc delete all --selector app.kubernetes.io/part-of=devworkspace-  
operator,app.kubernetes.io/name=devworkspace-webhook-server
```

```
$ oc delete serviceaccounts devworkspace-webhook-server -n openshift-operators
```

```
$ oc delete configmap devworkspace-controller -n openshift-operators
```

```
$ oc delete clusterrole devworkspace-webhook-server
```

```
$ oc delete clusterrolebinding devworkspace-webhook-server
```


5. Run the following commands to remove mutating or validating webhook configurations:

```
$ oc delete mutatingwebhookconfigurations controller.devfile.io
```

```
$ oc delete validatingwebhookconfigurations controller.devfile.io
```

6.3.2. Uninstalling the Operator using the web console

Procedure

1. In the **Administrator** perspective of the web console, navigate to **Operators → Installed Operators**.
2. Scroll the filter list or type a keyword into the **Filter by name** box to find the **Web Terminal Operator**.
3. Click the Options menu  for the Web Terminal Operator, and then select **Uninstall Operator**.
4. In the **Uninstall Operator** confirmation dialog box, click **Uninstall** to remove the Operator, Operator deployments, and pods from the cluster. The Operator stops running and no longer receives updates.

CHAPTER 7. DISABLING THE WEB CONSOLE IN OPENSIFT CONTAINER PLATFORM

You can disable the OpenShift Container Platform web console.

7.1. PREREQUISITES

- Deploy an OpenShift Container Platform cluster.

7.2. DISABLING THE WEB CONSOLE

You can disable the web console by editing the `consoles.operator.openshift.io` resource.

- Edit the `consoles.operator.openshift.io` resource:

```
$ oc edit consoles.operator.openshift.io cluster
```

The following example displays the parameters from this resource that you can modify:

```
apiVersion: operator.openshift.io/v1
kind: Console
metadata:
  name: cluster
spec:
  managementState: Removed 1
```

- 1** Set the `managementState` parameter value to **Removed** to disable the web console. The other valid values for this parameter are **Managed**, which enables the console under the cluster's control, and **Unmanaged**, which means that you are taking control of web console management.

CHAPTER 8. CREATING QUICK START TUTORIALS IN THE WEB CONSOLE

If you are creating quick start tutorials for the OpenShift Container Platform web console, follow these guidelines to maintain a consistent user experience across all quick starts.

8.1. UNDERSTANDING QUICK STARTS

A quick start is a guided tutorial with user tasks. In the web console, you can access quick starts under the **Help** menu. They are especially useful for getting oriented with an application, Operator, or other product offering.

A quick start primarily consists of tasks and steps. Each task has multiple steps, and each quick start has multiple tasks. For example:

- Task 1
 - Step 1
 - Step 2
 - Step 3
- Task 2
 - Step 1
 - Step 2
 - Step 3
- Task 3
 - Step 1
 - Step 2
 - Step 3

8.2. QUICK START USER WORKFLOW

When you interact with an existing quick start tutorial, this is the expected workflow experience:

1. In the **Administrator** or **Developer** perspective, click the **Help icon** and select **Quick Starts**.
2. Click a quick start card.
3. In the panel that appears, click **Start**.
4. Complete the on-screen instructions, then click **Next**.
5. In the **Check your work** module that appears, answer the question to confirm that you successfully completed the task.
 - a. If you select **Yes**, click **Next** to continue to the next task.

- b. If you select **No**, repeat the task instructions and check your work again.
6. Repeat steps 1 through 6 above to complete the remaining tasks in the quick start.
7. After completing the final task, click **Close** to close the quick start.

8.3. QUICK START COMPONENTS

A quick start consists of the following sections:

- **Card:** The catalog tile that provides the basic information of the quick start, including title, description, time commitment, and completion status
- **Introduction:** A brief overview of the goal and tasks of the quick start
- **Task headings:** Hyper-linked titles for each task in the quick start
- **Check your work module** A module for a user to confirm that they completed a task successfully before advancing to the next task in the quick start
- **Hints:** An animation to help users identify specific areas of the product
- **Buttons**
 - **Next and back buttons** Buttons for navigating the steps and modules within each task of a quick start
 - **Final screen buttons** Buttons for closing the quick start, going back to previous tasks within the quick start, and viewing all quick starts

The main content area of a quick start includes the following sections:

- **Card copy**
- **Introduction**
- **Task steps**
- **Modals and in-app messaging**
- **Check your work module**

8.4. CONTRIBUTING QUICK STARTS

OpenShift Container Platform introduces the quick start custom resource, which is defined by a **ConsoleQuickStart** object. Operators and administrators can use this resource to contribute quick starts to the cluster.

Prerequisites

- You must have cluster administrator privileges.

Procedure

1. To create a new quick start, run:

```
$ oc get -o yaml consolequickstart spring-with-s2i > my-quick-start.yaml
```

2. Run:

```
$ oc create -f my-quick-start.yaml
```

3. Update the YAML file using the guidance outlined in this documentation.

4. Save your edits.

8.4.1. Viewing the quick start API documentation

Procedure

- To see the quick start API documentation, run:

```
$ oc explain consolequickstarts
```

Run **oc explain -h** for more information about **oc explain** usage.

8.4.2. Mapping the elements in the quick start to the quick start CR

This section helps you visually map parts of the quick start custom resource (CR) with where they appear in the quick start within the web console.

8.4.2.1. conclusion element

Viewing the conclusion element in the YAML file

```
...
summary:
  failed: Try the steps again.
  success: Your Spring application is running.
title: Run the Spring application
conclusion: >-
  Your Spring application is deployed and ready. 1
```

1 conclusion text

Viewing the conclusion element in the web console

The conclusion appears in the last section of the quick start.

Get started with Spring 10 minutes



- 1 Create a Spring application
- 2 View the build status
- 3 View the associated Git repository
- 4 View the pod status
- 5 Change the deployment icon to Spring
- 6 Run the Spring application

Your Spring application is deployed and ready.

8.4.2.2. description element

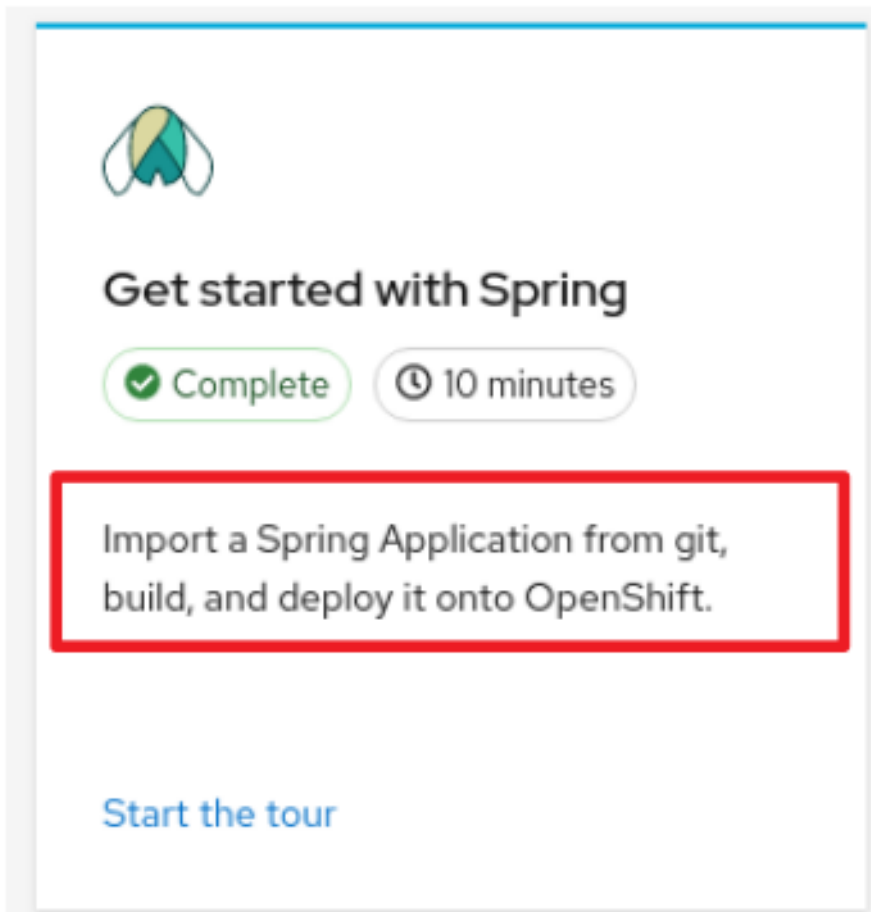
Viewing the description element in the YAML file

```
apiVersion: console.openshift.io/v1
kind: ConsoleQuickStart
metadata:
  name: spring-with-s2i
spec:
  description: 'Import a Spring Application from git, build, and deploy it onto OpenShift.' 1
  ...
```

- 1 description text

Viewing the description element in the web console

The description appears on the introductory tile of the quick start on the **Quick Starts** page.



8.4.2.3. displayName element

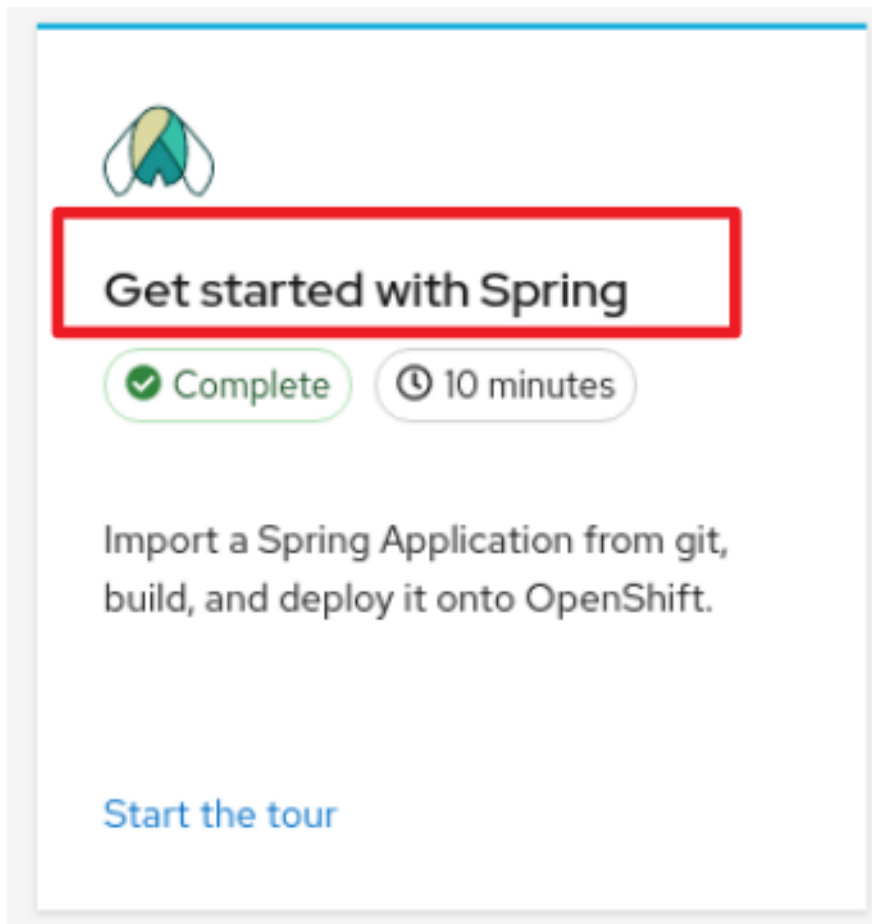
Viewing the displayName element in the YAML file

```
apiVersion: console.openshift.io/v1
kind: ConsoleQuickStart
metadata:
  name: spring-with-s2i
spec:
  description: 'Import a Spring Application from git, build, and deploy it onto OpenShift.'
  displayName: Get started with Spring 1
  durationMinutes: 10
```

1 **displayName** text.

Viewing the displayName element in the web console

The display name appears on the introductory tile of the quick start on the **Quick Starts** page.



8.4.2.4. durationMinutes element

Viewing the durationMinutes element in the YAML file

```

apiVersion: console.openshift.io/v1
kind: ConsoleQuickStart
metadata:
  name: spring-with-s2i
spec:
  description: 'Import a Spring Application from git, build, and deploy it onto OpenShift.'
  displayName: Get started with Spring
  durationMinutes: 10 1

```

- 1** **durationMinutes** value, in minutes. This value defines how long the quick start should take to complete.

Viewing the durationMinutes element in the web console

The duration minutes element appears on the introductory tile of the quick start on the **Quick Starts** page.


```

EsMTEwLTc5LjU3LDE0My40OC0xNTUuNiwwzLjxkLTguODgsNy45NS0xOC4wNSwxMi4yLTi3LjQzcTU
uNDIsOC41NCwxMS4zOSwxNi4yM2MzMS44NSw0MC45MSw3NS4xMiw2NC42NywxMzluMzIsNzluNj
NsMTguOCwyLjYyLDQuOTUtMTguMzNjMTMuMjYtNDkuMDcsMzUuMy05MC44NSw1MC42NC0xMT
YuMTksMTUuMzQsMjUuMzQsMzcuMzgsNjcuMTIsNTAuNjQsMTE2LjE5bDUUsMTguMzMsMTguOC0yL
jYyYzU3LjltOCwxMDAuNDctMzEuNzIsMTMyLjMyLTcyLjYzcTYtNy42OCwxMS4zOS0xNi4yM2M0Lj1L
DkuMzgsOC4yOSwxOC41NSwxMi4yLDI3LjQzLDmzLjQ5LDc2LDYyLjQyLDE0MS42OSwxNDMuNDgs
MTU1LjZsMS44MS4zMWgxLjg5YTYyLDIyLDAsMCwwLDE1LjU5LTYuNTJjNjMuMTUtNjQsMTAzLjk1LT
E0MC42LDExNC44OS0yMTUuNzhDMTAyNS43Myw2NjcuNjksMTAyMy4yOCw2MjkuMjIsMTAxMi42O
Sw1OTNali8+PHBhdGggY2xhc3M9ImNscy0yIiBkPSJNMzY0LjE1LDE4NS4yM2MxNy44OS0xNi40LDM
0LjctMzAuMTUsNDkuNzctNDAuMTFhMjE5LDIxMiwzLDAsMSw2NS45My0yNS43M0ExOTgsMTk4LDA
sMCwxLDUxMiwzMTYyMjdhMTk2LjExLDE5Ni4xMSwwLDAwMzswMiwzLjFjNC41LjkxLDkuMzYsMi4wN
wxNC41MywzLjUyLDYwLjQxLDIwLjQ4LDg0LjkyLDRkLjA1LTQ3LjQ0LDI0OC4wNi0yOC43NSwzNC4x
Mi0xNDAuNywxOTQuODQtMTg0LjY2LDEi2OC40MmE2MzAuODYsNjMwLjg2LDAsMCwwLTMzLjlyLD
U4LjMyQzI3Niww2NTUuMzQsMjY1LjQsNTk4LDEi2NS40LDUyMC4yOSwyNjUuNCwzNDAuNjEsMzExLjY
5LDI0MC43NCwzNjQuMTUsMTg1LjIzWiwvPjxwYXRolGNsYXNzPSJjbHhMjY0iTTUyNy41NCwzO
DQuODNjODQuMDYtOTkuNywxMTYyMDYtMTc3LjI4LDk1LjlyLTlzM3LjQzLjQ0LDE0OC4wNi0yOC43NSwzNC4x
E5LjIsMzcuMDYsMzEuMTMsNTUuNDgsNTUuNSw5OC43OCwxNTUuMzgsOTguNzgsMzM1LjA3LDAs
NzcuNzEtMTAuNiwxMzUuMDUtMjcuNzcsMTC3LjRhNjI4LjczLDYyOC43MywwLDAwMzswMjYyMy01OC
4zMmMjMzktNjUuMjYtMTMxLjQ1LDE5OS0xNzEuOTMtMjUyLjI3QzUyNi4zMywzODYyMjksNTI3LDM4
NS41Miw1MjcuNTQsMzg0LjgzWiwvPjxwYXRolGNsYXNzPSJjbHhMjY0iTTUyNy41NCwzO
DdoLS4wNmEuMzkuMzksMCwwLDEtLjI3Ls4xMWMtMTE5LjUyLDEyMS4wNy0xNTUuMTJg3LjQtNDcuN
TQtNDA0LjU4LDM0LjYzLTQxLjE0LDEyMC0xNTEuNiwyMDIuNzUtMjYyLjE5LTMuMTMsNy02LjE5LDE
0Lj1LTguOTIsMjEuNjktMjQuMzQsNjQuNDUtMzYyNjcsMTQ0LjMyLTM2LjY3LDIzNy40MSwwLDU2LjU
zLDUuNTgsMTA2LDE2LjU5LDE0Ny4xNEEzMDcuNDksMzA3LjQ5LDAwMzswLDI4MC45MSwzMjND
MjM3LDgxNi44OCwyMTYyOTMsODkzLjkzLDEzNC41OCw5MDguMDdali8+PHBhdGggY2xhc3M9ImN
scy01IiBkPSJNMzY0LjQzLDg0My43OUm1NjAuMTgsNzI3LjcyLDUxMiw2NjQuMTUsNTEyLDY2NC4xN
XMtNDguMTcsNjMuNTctNzEuNDMsMTQ5LjY0Yy00OC40NS02Ljc0LDEwMC45MS0yNy41Mi0xMzUu
NjYtOTEuMThhNjQ1LjY4LDY0NS42OCwwLDAwMzswOS41Ny03MS41NGwuMjEtLjMyLjE5LS4zM2M
zOC02My42MywxMjYyNC0xOTEuMzcsMTY3LjEyLTI0NS42Niww0MC43MSw1NC4yOCwxMjkuMSwzO
DIsMTY3LjE5LDE0NS42NmwuMTkuMzMuMjEuMzUuMzYyNjQ1LjY4LDY0NS42OCwwLDAwMzswOS41Nyw
3MS41NEM2ODQuMzQsNzg2LjI3LDYzMS44OCw4MdcuMDUsNTgzLjQzLDg0My43OVoilz48cGF0a
CBjbGFzc0iY2xzlTQiIGQ9Ik04ODkuNzUsOTA4YS4zOS4zOSwwLDAwMzswMjcuMTFoLS4wNkM4M
DcuMDcsODkzLjkzLDc4Nyw4MTYyODgsNzQzLjA5LDcyM2EzMDcuNDksMzA3LjQ5LDAwMzswLDEwL
jQ1LTU1LjU0YzExLTQxLjE5LDE2LjU5LTKwLjYxLDE2LjU5LTE0Ny4xNCwwLTKzLjA4LTYyLjMzLTE3M
y0zNi42Ni0yMzcuNHEtNC4yMi0xMS4xNi04LjkzLTlxLjdjODIuNzUsOTAuNTksMTY4LjE5LDEwMS4wNS
wyMDIuNzUsMjYyLjE5QzEwNDQuNzksNjIwLjU2LDEwMDkuMjcsNzg2Ljg5LDEwMS43NSw5MDhali8+
PC9zdmc+Cg==


```

...

- 1 The icon defined as a base64 value.

Viewing the icon element in the web console

The icon appears on the introductory tile of the quick start on the **Quick Starts** page.



Get started with Spring

✓ Complete
🕒 10 minutes

Import a Spring Application from git, build, and deploy it onto OpenShift.

[Start the tour](#)

8.4.2.6. introduction element

Viewing the introduction element in the YAML file

```

...
introduction: >- 1
  **Spring** is a Java framework for building applications based on a distributed microservices
  architecture.

  - Spring enables easy packaging and configuration of Spring applications into a self-contained
  executable application which can be easily deployed as a container to OpenShift.

  - Spring applications can integrate OpenShift capabilities to provide a natural "Spring on
  OpenShift" developer experience for both existing and net-new Spring applications. For example:

  - Externalized configuration using Kubernetes ConfigMaps and integration with Spring Cloud
  Kubernetes

  - Service discovery using Kubernetes Services

  - Load balancing with Replication Controllers

  - Kubernetes health probes and integration with Spring Actuator

  - Metrics: Prometheus, Grafana, and integration with Spring Cloud Sleuth

  - Distributed tracing with Istio & Jaeger tracing

  - Developer tooling through Red Hat OpenShift and Red Hat CodeReady developer tooling to

```

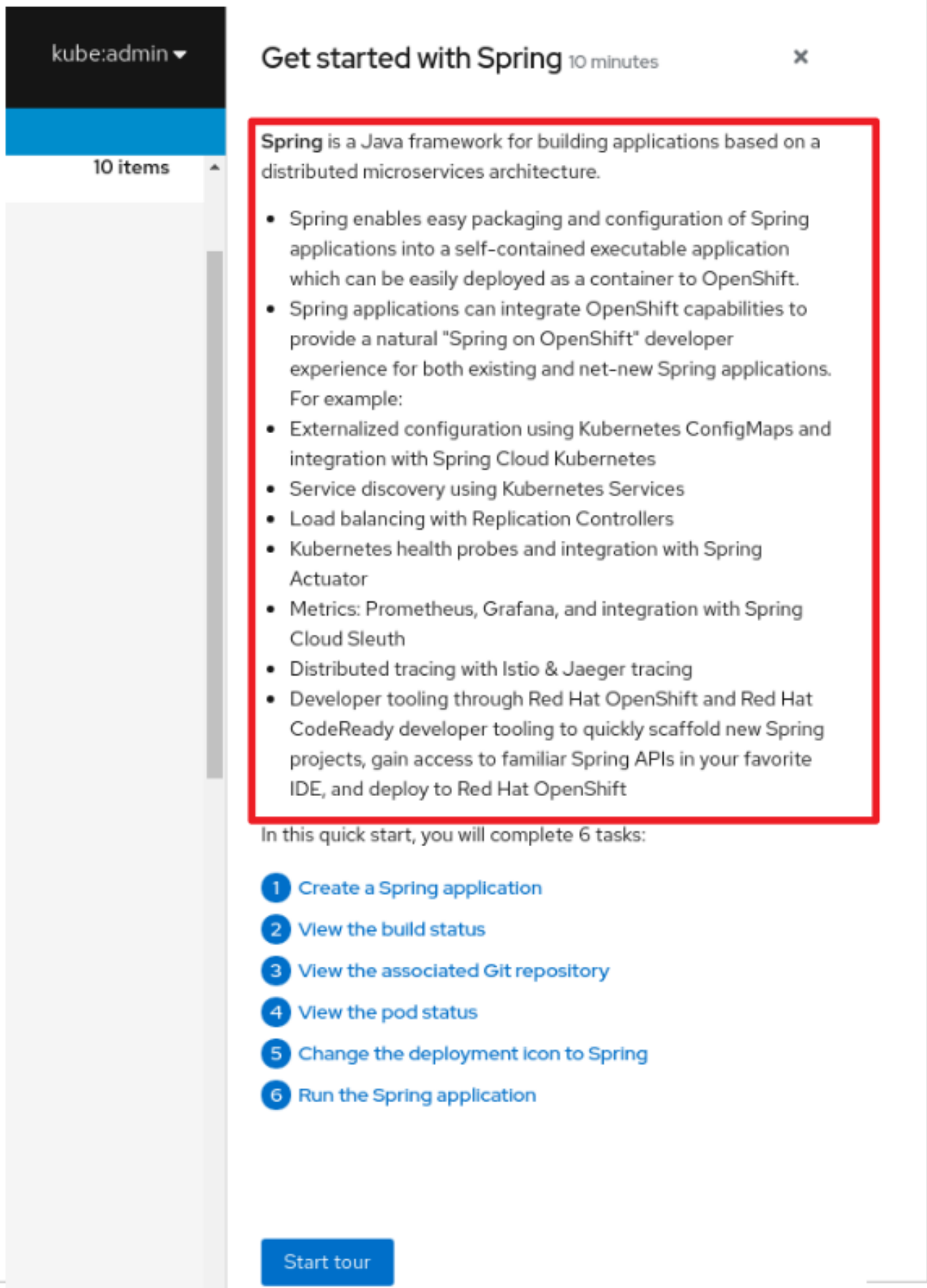
quickly scaffold new Spring projects, gain access to familiar Spring APIs in your favorite IDE, and deploy to Red Hat OpenShift

...

- 1 The introduction introduces the quick start and lists the tasks within it.

Viewing the introduction element in the web console

After clicking a quick start card, a side panel slides in that introduces the quick start and lists the tasks within it.



The screenshot shows the OpenShift web console interface. At the top left, there is a navigation menu with 'kube:admin' and a dropdown arrow. Below it, a blue bar indicates '10 items'. The main content area is titled 'Get started with Spring' with a duration of '10 minutes' and a close button (X). The content is enclosed in a red border. It starts with a paragraph: 'Spring is a Java framework for building applications based on a distributed microservices architecture.' This is followed by a bulleted list of features and integrations. Below the list, it says 'In this quick start, you will complete 6 tasks:' followed by a numbered list of six tasks. At the bottom, there is a blue button labeled 'Start tour'.

kube:admin ▾

Get started with Spring 10 minutes ✕

10 items ▲

Spring is a Java framework for building applications based on a distributed microservices architecture.

- Spring enables easy packaging and configuration of Spring applications into a self-contained executable application which can be easily deployed as a container to OpenShift.
- Spring applications can integrate OpenShift capabilities to provide a natural "Spring on OpenShift" developer experience for both existing and net-new Spring applications. For example:
 - Externalized configuration using Kubernetes ConfigMaps and integration with Spring Cloud Kubernetes
 - Service discovery using Kubernetes Services
 - Load balancing with Replication Controllers
 - Kubernetes health probes and integration with Spring Actuator
 - Metrics: Prometheus, Grafana, and integration with Spring Cloud Sleuth
 - Distributed tracing with Istio & Jaeger tracing
 - Developer tooling through Red Hat OpenShift and Red Hat CodeReady developer tooling to quickly scaffold new Spring projects, gain access to familiar Spring APIs in your favorite IDE, and deploy to Red Hat OpenShift

In this quick start, you will complete 6 tasks:

- 1 Create a Spring application
- 2 View the build status
- 3 View the associated Git repository
- 4 View the pod status
- 5 Change the deployment icon to Spring
- 6 Run the Spring application

[Start tour](#)

8.4.3. Adding a custom icon to a quick start

A default icon is provided for all quick starts. You can provide your own custom icon.

Next topic

Procedure

1. Find the **.svg** file that you want to use as your custom icon.
2. Use an [online tool to convert the text to base64](#) .
3. In the YAML file, add **icon: >-**, then on the next line include **data:image/svg+xml;base64** followed by the output from the base64 conversion. For example:

```
icon: >-
data:image/svg+xml;base64,PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdr
ciIHJvbGU9ImltZyldmld.
```

8.4.4. Limiting access to a quick start

Not all quick starts should be available for everyone. The **accessReviewResources** section of the YAML file provides the ability to limit access to the quick start.

To only allow the user to access the quick start if they have the ability to create **HelmChartRepository** resources, use the following configuration:

```
accessReviewResources:
- group: helm.openshift.io
  resource: helmchartrepositories
  verb: create
```

To only allow the user to access the quick start if they have the ability to list Operator groups and package manifests, thus ability to install Operators, use the following configuration:

```
accessReviewResources:
- group: operators.coreos.com
  resource: operatorgroups
  verb: list
- group: packages.operators.coreos.com
  resource: packagemanifests
  verb: list
```

8.4.5. Linking to other quick starts

Procedure

- In the **nextQuickStart** section of the YAML file, provide the **name**, not the **displayName**, of the quick start to which you want to link. For example:

```
nextQuickStart:
- add-healthchecks
```

8.4.6. Supported tags for quick starts

Write your quick start content in markdown using these tags. The markdown is converted to HTML.

Tag	Description
'b',	Defines bold text.
'img',	Embeds an image.
'i',	Defines italic text.
'strike',	Defines strike-through text.
's',	Defines smaller text
'del',	Defines smaller text.
'em',	Defines emphasized text.
'strong',	Defines important text.
'a',	Defines an anchor tag.
'p',	Defines paragraph text.
'h1',	Defines a level 1 heading.
'h2',	Defines a level 2 heading.
'h3',	Defines a level 3 heading.
'h4',	Defines a level 4 heading.
'ul',	Defines an unordered list.
'ol',	Defines an ordered list.
'li',	Defines a list item.
'code',	Defines a text as code.
'pre',	Defines a block of preformatted text.
'button',	Defines a button in text.

8.4.7. Quick start highlighting markdown reference

The highlighting, or hint, feature enables Quick Starts to contain a link that can highlight and animate a component of the web console.

The markdown syntax contains:

- Bracketed link text
- The **highlight** keyword, followed by the ID of the element that you want to animate

8.4.7.1. Perspective switcher

```
[Perspective switcher]{{highlight qs-perspective-switcher}}
```

8.4.7.2. Administrator perspective navigation links

```
[Home]{{highlight qs-nav-home}}
[Operators]{{highlight qs-nav-operators}}
[Workloads]{{highlight qs-nav-workloads}}
[Serverless]{{highlight qs-nav-serverless}}
[Networking]{{highlight qs-nav-networking}}
[Storage]{{highlight qs-nav-storage}}
[Service catalog]{{highlight qs-nav-servicecatalog}}
[Compute]{{highlight qs-nav-compute}}
[User management]{{highlight qs-nav-usermanagement}}
[Administration]{{highlight qs-nav-administration}}
```

8.4.7.3. Developer perspective navigation links

```
[Add]{{highlight qs-nav-add}}
[Topology]{{highlight qs-nav-topology}}
[Search]{{highlight qs-nav-search}}
[Project]{{highlight qs-nav-project}}
[Helm]{{highlight qs-nav-helm}}
```

8.4.7.4. Common navigation links

```
[Builds]{{highlight qs-nav-builds}}
[Pipelines]{{highlight qs-nav-pipelines}}
[Monitoring]{{highlight qs-nav-monitoring}}
```

8.4.7.5. Masthead links

```
[CloudShell]{{highlight qs-masthead-cloudshell}}
[Utility Menu]{{highlight qs-masthead-utilitymenu}}
[User Menu]{{highlight qs-masthead-usermenu}}
[Applications]{{highlight qs-masthead-applications}}
[Import]{{highlight qs-masthead-import}}
[Help]{{highlight qs-masthead-help}}
[Notifications]{{highlight qs-masthead-notifications}}
```

8.5. QUICK START CONTENT GUIDELINES

8.5.1. Card copy

You can customize the title and description on a quick start card, but you cannot customize the status.

- Keep your description to one to two sentences.
- Start with a verb and communicate the goal of the user. Correct example:

■ Create a serverless application.

8.5.2. Introduction

After clicking a quick start card, a side panel slides in that introduces the quick start and lists the tasks within it.

- Make your introduction content clear, concise, informative, and friendly.
- State the outcome of the quick start. A user should understand the purpose of the quick start before they begin.
- Give action to the user, not the quick start.
 - **Correct example:**

■ In this quick start, you will deploy a sample application to {product-title}.
 - **Incorrect example:**

■ This quick start shows you how to deploy a sample application to {product-title}.
- The introduction should be a maximum of four to five sentences, depending on the complexity of the feature. A long introduction can overwhelm the user.
- List the quick start tasks after the introduction content, and start each task with a verb. Do not specify the number of tasks because the copy would need to be updated every time a task is added or removed.
 - **Correct example:**

■ Tasks to complete: Create a serverless application; Connect an event source; Force a new revision
 - **Incorrect example:**

■ You will complete these 3 tasks: Creating a serverless application; Connecting an event source; Forcing a new revision

8.5.3. Task steps

After the user clicks **Start**, a series of steps appears that they must perform to complete the quick start.

Follow these general guidelines when writing task steps:

- Use "Click" for buttons and labels. Use "Select" for checkboxes, radio buttons, and drop-down menus.

- Use "Click" instead of "Click on"
 - **Correct example:**
Click OK.
 - **Incorrect example:**
Click on the OK button.
- Tell users how to navigate between **Administrator** and **Developer** perspectives. Even if you think a user might already be in the appropriate perspective, give them instructions on how to get there so that they are definitely where they need to be.
Examples:
 - Enter the Developer perspective: In the main navigation, click the dropdown menu and select Developer.
 - Enter the Administrator perspective: In the main navigation, click the dropdown menu and select Admin.
- Use the "Location, action" structure. Tell a user where to go before telling them what to do.
 - **Correct example:**
In the node.js deployment, hover over the icon.
 - **Incorrect example:**
Hover over the icon in the node.js deployment.
- Keep your product terminology capitalization consistent.
- If you must specify a menu type or list as a dropdown, write "dropdown" as one word without a hyphen.
- Clearly distinguish between a user action and additional information on product functionality.
 - **User action:**
Change the time range of the dashboard by clicking the dropdown menu and selecting time range.
 - **Additional information:**
To look at data in a specific time frame, you can change the time range of the dashboard.
- Avoid directional language, like "In the top-right corner, click the icon". Directional language becomes outdated every time UI layouts change. Also, a direction for desktop users might not be accurate for users with a different screen size. Instead, identify something using its name.
 - **Correct example:**
In the navigation menu, click Settings.

- **Incorrect example:**
 - █ In the left-hand menu, click Settings.
- Do not identify items by color alone, like "Click the gray circle". Color identifiers are not useful for sight-limited users, especially colorblind users. Instead, identify an item using its name or copy, like button copy.
 - **Correct example:**
 - █ The success message indicates a connection.
 - **Incorrect example:**
 - █ The message with a green icon indicates a connection.
- Use the second-person point of view, you, consistently:
 - **Correct example:**
 - █ Set up your environment.
 - **Incorrect example:**
 - █ Let's set up our environment.

8.5.4. Check your work module

- After a user completes a step, a **Check your work** module appears. This module prompts the user to answer a yes or no question about the step results, which gives them the opportunity to review their work. For this module, you only need to write a single yes or no question.
 - If the user answers **Yes**, a check mark will appear.
 - If the user answers **No**, an error message appears with a link to relevant documentation, if necessary. The user then has the opportunity to go back and try again.

8.5.5. Formatting UI elements

Format UI elements using these guidelines:

- Copy for buttons, dropdowns, tabs, fields, and other UI controls: Write the copy as it appears in the UI and bold it.
- All other UI elements—including page, window, and panel names: Write the copy as it appears in the UI and bold it.
- Code or user-entered text: Use monospaced font.
- Hints: If a hint to a navigation or masthead element is included, style the text as you would a link.
- CLI commands: Use monospaced font.
- In running text, use a bold, monospaced font for a command.

- If a parameter or option is a variable value, use an italic monospaced font.
- Use a bold, monospaced font for the parameter and a monospaced font for the option.

8.6. ADDITIONAL RESOURCES

- For voice and tone requirements, refer to [PatternFly's brand voice and tone guidelines](#).
- For other UX content guidance, refer to all areas of [PatternFly's UX writing style guide](#).