OpenShift Container Platform 4.11

Web console

Getting started with the web console in OpenShift Container Platform
Getting started with the web console in OpenShift Container Platform
Abstract

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

CHAPTER 1. WEB CONSOLE OVERVIEW .................................................................................. 8
  1.1. ABOUT THE ADMINISTRATOR PERSPECTIVE IN THE WEB CONSOLE  8
    1.1.1. Accessing the Administrator perspective  8
  1.2. ABOUT THE DEVELOPER PERSPECTIVE IN THE WEB CONSOLE  9
    1.2.1. Accessing the Developer perspective  9

CHAPTER 2. ACCESSING THE WEB CONSOLE ..................................................................... 12
  2.1. PREREQUISITES  12
  2.2. UNDERSTANDING AND ACCESSING THE WEB CONSOLE  12
  2.3. MULTICLUSTER CONSOLE
    2.3.1. Enabling multicluster in the web console  12

CHAPTER 3. USING THE OPENShift CONTAINER PLATFORM DASHBOARD TO GET CLUSTER INFORMATION ........................................... 15
  3.1. ABOUT THE OPENShift CONTAINER PLATFORM DASHBOARDS PAGE  15

CHAPTER 4. ADDING USER PREFERENCES ....................................................................... 17
  4.1. SETTING USER PREFERENCES  17

CHAPTER 5. CONFIGURING THE WEB CONSOLE IN OPENShift CONTAINER PLATFORM .......... 18
  5.1. PREREQUISITES  18
  5.2. CONFIGURING THE WEB CONSOLE  18

CHAPTER 6. CUSTOMIZING THE WEB CONSOLE IN OPENShift CONTAINER PLATFORM .......... 19
  6.1. ADDING A CUSTOM LOGO AND PRODUCT NAME  19
  6.2. CREATING CUSTOM LINKS IN THE WEB CONSOLE  20
  6.3. CUSTOMIZING CONSOLE ROUTES
    6.3.1. Customizing the console route  21
    6.3.2. Customizing the download route  22
  6.4. CUSTOMIZING THE LOGIN PAGE  23
  6.5. DEFINING A TEMPLATE FOR AN EXTERNAL LOG LINK  24
  6.6. CREATING CUSTOM NOTIFICATION BANNERS  25
  6.7. CUSTOMIZING CLI DOWNLOADS  25
  6.8. ADDING YAML EXAMPLES TO KUBERNETES RESOURCES  26

CHAPTER 7. ADDING A DYNAMIC PLUG-IN TO THE OPENShift CONTAINER PLATFORM WEB CONSOLE .......................................................... 28
  7.1. ABOUT DYNAMIC PLUG-INS  28
    7.1.1. Key features  28
    7.1.2. PatternFly 4 guidelines  28
    7.1.3. General guidelines  28
  7.2. ENABLE DYNAMIC PLUG-INS IN THE WEB CONSOLE  29
  7.3. GETTING STARTED WITH DYNAMIC PLUG-INS  29
  7.4. RUNNING YOUR DYNAMIC PLUG-IN  30
  7.5. ADDING A NEW EXTENSION TO YOUR PLUG-IN
    7.5.1. Dynamic plug-in extension types
      7.5.1.1. console.action/filter  31
      7.5.1.1.1. Summary  31
      7.5.1.1.2. Properties  32
      7.5.1.2. console.action/group  32
      7.5.1.2.1. Summary  32
      7.5.1.2.2. Properties  32
      7.5.1.3. console.action/provider  33
7.5.1.38.2. Properties
7.5.1.39. console.page/resource/details
  7.5.1.39.1. Summary
  7.5.1.39.2. Properties
7.5.1.40. console.page/resource/list
  7.5.1.40.1. Summary
  7.5.1.40.2. Properties
7.5.1.41. console.page/route
  7.5.1.41.1. Summary
  7.5.1.41.2. Properties
7.5.1.42. console.page/route/standalone
  7.5.1.42.1. Summary
  7.5.1.42.2. Properties
7.5.1.43. console.perspective
  7.5.1.43.1. Summary
  7.5.1.43.2. Properties
7.5.1.44. console.project-overview/inventory-item
  7.5.1.44.1. Summary
  7.5.1.44.2. Properties
7.5.1.45. console.project-overview/utilization-item
  7.5.1.45.1. Summary
  7.5.1.45.2. Properties
7.5.1.46. console.pvc/alert
  7.5.1.46.1. Summary
  7.5.1.46.2. Properties
7.5.1.47. console.pvc/create-prop
  7.5.1.47.1. Summary
  7.5.1.47.2. Properties
7.5.1.48. console.pvc/delete
  7.5.1.48.1. Summary
  7.5.1.48.2. Properties
7.5.1.49. console.pvc/status
  7.5.1.49.1. Summary
  7.5.1.49.2. Properties
7.5.1.50. console.redux-reducer
  7.5.1.50.1. Summary
  7.5.1.50.2. Properties
7.5.1.51. console.resource/create
  7.5.1.51.1. Summary
  7.5.1.51.2. Properties
7.5.1.52. console.storage-provider
  7.5.1.52.1. Summary
  7.5.1.52.2. Properties
7.5.1.53. console.tab/horizontalNav
  7.5.1.53.1. Summary
  7.5.1.53.2. Properties
7.5.1.54. console.telemetry/listener
  7.5.1.54.1. Summary
  7.5.1.54.2. Properties
7.5.1.55. console.topology/adapter/build
  7.5.1.55.1. Summary
  7.5.1.55.2. Properties
7.5.1.56. console.topology/adapter/network
7.5.1.56.1. Summary 62
7.5.1.56.2. Properties 63
7.5.1.57. console.topology/adapter/pod 63
  7.5.1.57.1. Summary 63
  7.5.1.57.2. Properties 63
7.5.1.58. console.topology/component/factory 63
  7.5.1.58.1. Summary 63
  7.5.1.58.2. Properties 63
7.5.1.59. console.topology/create/connector 63
  7.5.1.59.1. Summary 63
  7.5.1.59.2. Properties 63
7.5.1.60. console.topology/data/factory 64
  7.5.1.60.1. Summary 64
  7.5.1.60.2. Properties 64
7.5.1.61. console.topology/decorator/provider 64
  7.5.1.61.1. Summary 64
  7.5.1.61.2. Properties 64
7.5.1.62. console.topology/details/resource-alert 65
  7.5.1.62.1. Summary 65
  7.5.1.62.2. Properties 65
7.5.1.63. console.topology/details/resource-link 65
  7.5.1.63.1. Summary 65
  7.5.1.63.2. Properties 65
7.5.1.64. console.topology/details/tab 66
  7.5.1.64.1. Summary 66
  7.5.1.64.2. Properties 66
7.5.1.65. console.topology/details/tab-section 66
  7.5.1.65.1. Summary 66
  7.5.1.65.2. Properties 66
7.5.1.66. console.topology/display/filters 67
  7.5.1.66.1. Summary 67
  7.5.1.66.2. Properties 67
7.5.1.67. console.topology/relationship/provider 68
  7.5.1.67.1. Summary 68
  7.5.1.67.2. Properties 68
7.5.1.68. console.user-preference/group 68
  7.5.1.68.1. Summary 68
  7.5.1.68.2. Properties 68
7.5.1.69. console.user-preference/item 68
  7.5.1.69.1. Summary 68
  7.5.1.69.2. Properties 69
7.5.1.70. console.yaml-template 69
  7.5.1.70.1. Summary 69
  7.5.1.70.2. Properties 69
7.5.1.71. dev-console.add/action 70
  7.5.1.71.1. Summary 70
  7.5.1.71.2. Properties 70
7.5.1.72. dev-console.add/action-group 70
  7.5.1.72.1. Summary 70
  7.5.1.72.2. Properties 71
7.5.1.73. dev-console.import/environment 71
  7.5.1.73.1. Summary 71
  7.5.1.73.2. Properties 71
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](image)

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
- Overview of the Administrator perspective
- Creating and deploying applications on OpenShift Container Platform using the Developer perspective
• Viewing the applications in your project, verifying their deployment status, and interacting with them in the Topology view

• 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. After 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.

2.3. MULTICLUSTER CONSOLE

The multicluster console provides a single interface with consistent design for the hybrid cloud console. If you enable the feature, you can switch between Red Hat Advanced Cluster Management (RHACM) and the cluster console in the same browser tab. It provides a simplified and consistent design that allows for shared components.

2.3.1. Enabling multicluster in the web console
IMPORTANT

Multicluster console 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/.

Prerequisites

- Your cluster must be using the latest version of OpenShift Container Platform.
- You must have Red Hat Advanced Cluster Management (RHACM) or the multicluster engine for Kubernetes (MCE) installed.
- You must have administrator privileges.

WARNING

Do not set this feature gate on production clusters. You will not be able to upgrade your cluster after applying the feature gate, and it cannot be undone.

Procedure

1. Log in to the OpenShift Container Platform web console using your credentials.

2. Enable RHACM in the administrator perspective by navigating from Administration → Cluster Settings → Configuration → Console console.operator.openshift.io → Console Plugins and click Enable for acm.

3. A pop-up window will appear notifying you that updating the enablement of this console plug-in will prompt for the console to be refreshed once it has been updated. Select Enable and click Save.

4. Repeat the previous two steps for the mce console plug-in immediately after enabling acm.

5. A pop-up window that states that a web console update is available will appear a few moments after you enable. Click Refresh the web console in the pop-up window to update.

NOTE

You might see the pop-up window to refresh the web console twice if the second redeployment has not occurred by the time you click Refresh the web console.

- local-cluster and All Clusters is now visible above the perspectives in the navigation section.
6. Enable the feature gate by navigating from Administration → Cluster Settings → Configuration → FeatureGate, and edit the YAML template as follows:

```yaml
   spec:
      featureSet: TechPreviewNoUpgrade
```

7. Click Save to enable the multicloud console for all clusters.

**IMPORTANT**

After you save, this feature is enabled and cannot be undone.

Additional resources

- Enabling feature sets using the web console
CHAPTER 3. USING THE OPENSHIFT 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 OPENSHIFT 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. ADDING USER PREFERENCES

You can change the default preferences for your profile to meet your requirements. You can set your default project, topology view (graph/list), editing medium (form or YAML), and language preferences.

The changes made to the user preferences are automatically saved.

4.1. SETTING USER PREFERENCES

You can set the default user preferences for your cluster.

Procedure

1. Log in to the OpenShift Container Platform web console using your login credentials.

2. Use the masthead to access the user preferences under the user profile.

3. In the General section:
   a. In the Perspective field, you can set the default perspective you want to be logged in to. You can select the Administrator or the Developer perspective as required. If a perspective is not selected, you are logged into the perspective you last visited.
   b. In the Project field, select a project you want to work in. The console will default to the project every time you log in.
   c. In the Topology field, you can set the topology view to default to the graph or list view. If not selected, the console defaults to the last view you used.
   d. In the Create/Edit resource method field, you can set a preference for creating or editing a resource. If both the form and YAML options are available, the console defaults to your selection.

4. In the Language section, select Default browser language to use the default browser language settings. Otherwise, select the language that you want to use for the console.
CHAPTER 5. CONFIGURING THE WEB CONSOLE IN OPENSHIFT CONTAINER PLATFORM

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

5.1. PREREQUISITES

- Deploy an OpenShift Container Platform cluster.

5.2. 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:

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

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

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

  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 6. CUSTOMIZING THE WEB CONSOLE IN OPENSHIFT 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.

6.1. 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
   ```

   **TIP**

   You can alternatively apply the following YAML to create the config map:

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

   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
   ```

   ```yaml
   apiVersion: operator.openshift.io/v1
   kind: Console
   metadata:
     name: cluster
   spec:
     customization:
       customLogoFile:
   ```
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 console Operator configuration, run:

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

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

```yaml
apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: namespaced-dashboard-link-for-all-namespaces
spec:
```

### 6.2. 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:

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

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

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

```yaml
apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: namespaced-dashboard-link-for-all-namespaces
spec:
```
To make the custom link appear in only some namespaces, follow this example:

```yaml
apiVersion: console.openshift.io/v1
kind: ConsoleLink
metadata:
  name: namespaced-dashboard-for-some-namespaces
spec:
  href: 'https://www.example.com'
  location: NamespaceDashboard
  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:

```yaml
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.

---

### 6.3. CUSTOMIZING CONSOLE ROUTES

For **console** and **downloads** routes, custom routes functionality uses the **ingress** config route configuration API. If the **console** custom route is set up in both the **ingress** config and **console-operator** config, then the new **ingress** config custom route configuration takes precedence. The route configuration with the **console-operator** config is deprecated.

#### 6.3.1. Customizing the console route

You can customize the console route by setting the custom hostname and TLS certificate in the **spec.componentRoutes** field of the cluster **Ingress** configuration.

**Prerequisites**
You have logged in to the cluster as a user with administrative privileges.

You have created a secret in the `openshift-config` namespace containing the TLS certificate and key. This is required if the domain for the custom hostname suffix does not match the cluster domain suffix. The secret is optional if the suffix matches.

TIP

You can create a TLS secret by using the `oc create secret tls` command.

Procedure

1. Edit the cluster `Ingress` configuration:

   $ oc edit ingress.config.openshift.io cluster

2. Set the custom hostname and optionally the serving certificate and key:

   ```
   apiVersion: config.openshift.io/v1
   kind: Ingress
   metadata:
     name: cluster
   spec:
     componentRoutes:
     - name: console
       namespace: openshift-console
       hostname: <custom_hostname>
       servingCertKeyPairSecret:
         name: <secret_name>
   ```

   1. The custom hostname.
   2. Reference to a secret in the `openshift-config` namespace that contains a TLS certificate (`tls.crt`) and key (`tls.key`). This is required if the domain for the custom hostname suffix does not match the cluster domain suffix. The secret is optional if the suffix matches.

3. Save the file to apply the changes.

6.3.2. Customizing the download route

You can customize the download route by setting the custom hostname and TLS certificate in the `spec.componentRoutes` field of the cluster `Ingress` configuration.

Prerequisites

- You have logged in to the cluster as a user with administrative privileges.
- You have created a secret in the `openshift-config` namespace containing the TLS certificate and key. This is required if the domain for the custom hostname suffix does not match the cluster domain suffix. The secret is optional if the suffix matches.
**TIP**

You can create a TLS secret by using the `oc create secret tls` command.

**Procedure**

1. Edit the cluster **Ingress** configuration:

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

2. Set the custom hostname and optionally the serving certificate and key:

   ```
   apiVersion: config.openshift.io/v1
   kind: Ingress
   metadata:
     name: cluster
   spec:
     componentRoutes:
     - name: downloads
       namespace: openshift-console
       hostname: <custom_hostname>  
       servingCertKeyPairSecret:
         name: <secret_name>  
   ```

   **1** The custom hostname.

   **2** Reference to a secret in the `openshift-config` namespace that contains a TLS certificate (`tls.crt`) and key (`tls.key`). This is required if the domain for the custom hostname suffix does not match the cluster domain suffix. The secret is optional if the suffix matches.

3. Save the file to apply the changes.

### 6.4. 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 htpasswd.

**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:
   ```yaml
   spec:
     templates:
       error:
         name: error-template
       login:
         name: login-template
       providerSelection:
         name: providers-template
   ```
   Run `oc explain oauths.spec.templates` to understand the options.

### 6.5. 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:

   ```yaml
   apiVersion: console.openshift.io/v1
   kind: ConsoleExternalLogLink
   metadata:
   ```
6.6. CREATING CUSTOM NOTIFICATION BANNERS

**Prerequisites**

- You must have administrator privileges.

**Procedure**

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

   ```yaml
   name: example
   spec:
     hrefTemplate: >-
       https://example.com/logs?
       resourceName=${resourceName}&containerName=${containerName}&resourceNamespace=${resourceNamespace}&podLabels=${podLabels}
     text: Example Logs
   apiVersion: console.openshift.io/v1
   kind: ConsoleNotification
   metadata:
     name: example
   spec:
     text: This is an example notification message with an optional link.
     location: BannerTop
     link:
       href: 'https://www.example.com'
       text: Optional link text
       color: '#fff'
       backgroundColor: '#0088ce'
   ``

   Valid location settings are **BannerTop**, **BannerBottom**, and **BannerTopBottom**.
4. Click Create to apply your changes.

6.7. 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**

1. 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:

```yaml
apiVersion: console.openshift.io/v1
class: ConsoleCLIDownload
metadata:
  name: example-cli-download-links-for-foo
spec:
  description: |
    This is an example of download links for foo
displayName: example-foo
links:
  text: foo for linux
  text: foo for mac
  text: foo for windows
```

4. Click the Save button.

### 6.8. 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:

```yaml
apiVersion: console.openshift.io/v1
class: 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:
```

OpenShift Container Platform 4.11 Web console
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 7. ADDING A DYNAMIC PLUG-IN TO THE OPENSHIFT CONTAINER PLATFORM WEB CONSOLE

You can create and deploy a dynamic plug-in on your cluster that is loaded at run-time.

IMPORTANT

Creating a dynamic plug-in 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/.

7.1. ABOUT DYNAMIC PLUG-INS

A dynamic plug-in allows you to add custom pages and other extensions to your interface at runtime. The ConsolePlugin custom resource registers plug-ins with the console, and a cluster administrator enables plug-ins in the console-operator configuration.

7.1.1. Key features

A dynamic plug-in allows you to make the following customizations to the OpenShift Container Platform experience:

- Add custom pages.
- Add perspectives and update navigation items.
- Add tabs and actions to resource pages.

7.1.2. PatternFly 4 guidelines

When creating your plug-in, follow these guidelines for using PatternFly:

- Use PatternFly4 components and PatternFly CSS variables. Core PatternFly components are available through the SDK. Using PatternFly components and variables will help your plug-in look consistent in future console versions.
- Make your plug-in accessible by following PatternFly’s accessibility fundamentals.
- Do not use other CSS libraries such as Bootstrap or Tailwind. They can conflict with PatternFly and will not match the console look and feel.

7.1.3. General guidelines

When creating your plug-in, follow these general guidelines:

- Prefix your CSS class names with your plug-in name to avoid collisions. For example, my-plugin__heading and my-plugin__icon.
- Maintain a consistent look, feel, and behavior with other console pages.
Follow `react-i18next` localization guidelines when creating your plug-in. You can use the `useTranslation` hook like the one in the following example:

```javascript
const Header: React.FC = () => {
  const { t } = useTranslation('plugin__console-demo-plugin');
  return <h1>{t('Hello, World!')}</h1>;
};
```

Do not use console CSS classes in your markup or override console CSS classes. These are not APIs and are subject to change. Using them might break your plug-in. Avoid selectors like element selectors that could affect markup outside of your plug-in’s components.

### 7.2. ENABLE DYNAMIC PLUG-INS IN THE WEB CONSOLE

Cluster administrators can enable plug-ins in the web console browser. Dynamic plug-ins are disabled by default. In order to enable, a cluster administrator will need to enable them in the `console-operator` configuration.

**Procedure**

1. In the Administration → Cluster Settings page of the web console, click the Configuration tab.
2. Click the `Console operator.openshift.io` configuration resource.
3. From there, click the Console plugins tab to view the dynamic plug-ins running.
4. In the Status column, click Enable console plugin in the pop-over menu, which will launch the Console plugin enablement modal.
5. Click Enable and Save.

**Verification**

- Refresh the browser to view the enabled plug-in.

### 7.3. GETTING STARTED WITH DYNAMIC PLUG-INS

To get started using the dynamic plug-in, you must set up your environment to write a new OpenShift Console dynamic plug-in.

**Prerequisites**

- Ensure you have Node.js installed.
- Ensure you have yarn installed.

**Procedure**

1. Edit the plug-in metadata in the `consolePlugin` declaration of `package.json`.

```json
"consolePlugin": {
  "name": "my-plugin",  
  "version": "0.0.1", 
} 
```
"displayName": "My Plugin",  
"description": "Enjoy this shiny, new console plugin!",  
"exposedModules": { 
  "ExamplePage": "./components/ExamplePage" 
},  
"dependencies": { 
  "@console/pluginAPI": "***" 
} 

1. Update the name of your plug-in.  
2. Update the version.  
3. Update the display name for your plug-in.  
4. Update the description with a synopsis about your plug-in. 

7.4. RUNNING YOUR DYNAMIC PLUG-IN

You can run the plug-in using a local development environment. The OpenShift Container Platform web console runs in a container connected to the cluster you have logged into.

Prerequisites

- You must have the OpenShift CLI (`oc`) installed.
- You must have an OpenShift cluster running.
- You must have Docker or at least v3.2.0 of Podman installed.

Procedure

1. Build a plug-in and generate the output to the `dist` directory by running

   $ yarn build

2. Start an HTTP server by running

   $ yarn http-server

3. The HTTP server, which runs on port 9001, generates the following assets with caching disabled and CORS enabled.

   Starting up http-server, serving ./dist
   Available on:
   http://127.0.0.1:9001
   http://192.168.1.190:9001
   http://10.40.192.80:9001
   Hit CTRL-C to stop the server

4. Optional: Add additional server options to the script by running
$ yarn http-server -a <server name>

5. Direct **bridge** to proxy requests to your local plug-in asset server by running

```bash
$ ./bin/bridge -plugins console-demo-plugin=http://localhost:9001/
```

**Verification**

- Visit **local host** to view the running plug-in. Inspect the value of `window.SERVER_FLAGS.consolePlugins` to see the list of plug-ins which load at runtime.

### 7.5. ADDING A NEW EXTENSION TO YOUR PLUG-IN

You can add extensions that allow you to customize your plug-in. Those extensions are then loaded to the console at run-time.

1. Edit the `console-extensions.json` file:

   ```json
   [
   {
   "type": "console.flag",
   "properties": {
   "handler": {
   "$codeRef": "barUtils.testHandler"
   }
   },
   {
   "type": "console.flag/model",
   "properties": {
   "flag": "EXAMPLE",
   "model": {
   "group": "kubevirt.io",
   "version": "v1alpha3",
   "kind": "ExampleModel"
   }
   }
   }
   ]
   ```

   **1** Add the extension type(s) you want to include with this plug-in. You can include multiple extensions separated with a comma.

   **2** The `$codeRef` value should be formatted as either `moduleName.exportName` for a named export or `moduleName` for the default export. Only the plug-in’s exported modules can be used in code references.

### 7.5.1. Dynamic plug-in extension types

#### 7.5.1.1. console.action/filter

**Summary**

`ActionFilter` can be used to filter an action.
7.5.1.1.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>string</td>
<td>no</td>
<td>The context ID helps to narrow the scope of contributed actions to a particular area of the application. Examples include <strong>topology</strong> and <strong>helm</strong>.</td>
</tr>
<tr>
<td>filter</td>
<td>CodeRef&lt;scope: any, action: Action&gt; ⇒ boolean&gt;</td>
<td>no</td>
<td>A function that will filter actions based on some conditions. <strong>scope</strong>: The scope in which actions should be provided for. A hook might be required if you want to remove the <strong>ModifyCount</strong> action from a deployment with a horizontal pod autoscaler (HPA).</td>
</tr>
</tbody>
</table>

7.5.1.2. console.action/group

7.5.1.2.1. Summary

**ActionGroup** contributes an action group that can also be a submenu.

7.5.1.2.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>ID used to identify the action section.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>yes</td>
<td>The label to display in the UI. Required for submenus.</td>
</tr>
<tr>
<td>submenu</td>
<td>boolean</td>
<td>yes</td>
<td>Whether this group should be displayed as submenu.</td>
</tr>
</tbody>
</table>
7.5.1.3. console.action/provider

7.5.1.3.1. Summary

**ActionProvider** contributes a hook that returns list of actions for specific context.

7.5.1.3.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextId</td>
<td>string</td>
<td>no</td>
<td>The context ID helps to narrow the scope of contributed actions to a particular area of the application. Examples include <code>topology</code> and <code>helm</code>.</td>
</tr>
<tr>
<td>provider</td>
<td>CodeRef&lt;Extension Hook&lt;Action[], any&gt;&gt;</td>
<td>no</td>
<td>A React hook that returns actions for the given scope. If <code>contextId = resource</code>, then the scope will always be a Kubernetes resource object.</td>
</tr>
</tbody>
</table>

7.5.1.4. console.action/resource-provider

7.5.1.4.1. Summary

**ResourceActionProvider** contributes a hook that returns list of actions for specific resource model.

7.5.1.4.2. Properties
<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ExtensionK8sKindVersionModel</td>
<td>no</td>
<td>The model for which this provider provides actions for.</td>
</tr>
<tr>
<td>provider</td>
<td>CodeRef&lt;Extension Hook&lt;Action[], any&gt;&gt;</td>
<td>no</td>
<td>A react hook which returns actions for the given resource model</td>
</tr>
</tbody>
</table>

7.5.1.5. console.alert-action

7.5.1.5.1. Summary

(not available)

7.5.1.5.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>string</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>text</td>
<td>string</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td>CodeRef&lt;(alert: any) ⇒ void&gt;</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

7.5.1.6. console.catalog/item-filter

7.5.1.6.1. Summary

(not available)

7.5.1.6.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogId</td>
<td>string</td>
<td>string</td>
<td>The unique identifier for the catalog this provider contributes to.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>no</td>
<td>Type ID for the catalog item type.</td>
</tr>
</tbody>
</table>
7.5.1.7. console.catalog/item-metadata

7.5.1.7.1. Summary

(not available)

7.5.1.7.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogId</td>
<td>string</td>
<td>no</td>
<td>The unique identifier for the catalog this provider contributes to.</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>no</td>
<td>Type ID for the catalog item type.</td>
</tr>
<tr>
<td>provider</td>
<td>CodeRef&lt;Extension Hook&lt;CatalogItemMetadataProviderFunction, CatalogExtensionHookOptions&gt;&gt;</td>
<td>no</td>
<td>A hook which returns a function that will be used to provide metadata to catalog items of a specific type.</td>
</tr>
</tbody>
</table>

7.5.1.8. console.catalog/item-provider

7.5.1.8.1. Summary

(not available)

7.5.1.8.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>catalogId</td>
<td>string</td>
<td>no</td>
<td>The unique identifier for the catalog this provider contributes to.</td>
</tr>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>no</td>
<td>Type ID for the catalog item type.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>Title for the catalog item.</td>
</tr>
<tr>
<td>provider</td>
<td>CodeRef&lt;Extension Hook&lt;CatalogItem&lt;any&gt;[]&gt;, CatalogExtensionHookOptions&gt;&gt;</td>
<td>no</td>
<td>Fetch items and normalize it for the catalog. Value is a react effect hook.</td>
</tr>
<tr>
<td>priority</td>
<td>number</td>
<td>yes</td>
<td>Priority for this provider. Defaults to 0. Higher priority providers may override catalog items provided by other providers.</td>
</tr>
</tbody>
</table>

### 7.5.1.9. console.catalog/item-type

#### 7.5.1.9.1. Summary

(not available)

#### 7.5.1.9.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>no</td>
<td>Type for the catalog item.</td>
</tr>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>Title for the catalog item.</td>
</tr>
<tr>
<td>catalogDescription</td>
<td>string</td>
<td>CodeRef&lt;React.ReactNode&gt;</td>
<td>yes</td>
</tr>
<tr>
<td>typeDescription</td>
<td>string</td>
<td>yes</td>
<td>Description for the catalog item type.</td>
</tr>
<tr>
<td>filters</td>
<td>CatalogItemAttribute[]</td>
<td>yes</td>
<td>Custom filters specific to the catalog item.</td>
</tr>
</tbody>
</table>
7.5.1.10. `console.catalog/item-type-metadata`

7.5.1.10.1. Summary

(not available)

7.5.1.10.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>string</td>
<td>no</td>
<td>Type for the catalog item.</td>
</tr>
<tr>
<td>filters</td>
<td><code>CatalogItemAttribute</code> []</td>
<td>yes</td>
<td>Custom filters specific to the catalog item.</td>
</tr>
<tr>
<td>groupings</td>
<td><code>CatalogItemAttribute</code> []</td>
<td>yes</td>
<td>Custom groupings specific to the catalog item.</td>
</tr>
</tbody>
</table>

7.5.1.11. `console.cluster-overview/inventory-item`

7.5.1.11.1. Summary

Adds a new inventory item into cluster overview page.

7.5.1.11.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td><code>CodeRef&lt;React.ComponentType&lt;{}&gt;&gt;</code></td>
<td>no</td>
<td>The component to be rendered.</td>
</tr>
</tbody>
</table>

7.5.1.12. `console.cluster-overview/multiline-utilization-item`

7.5.1.12.1. Summary

Adds a new cluster overview multi-line utilization item.

7.5.1.12.2. Properties
<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>The title of the utilization item.</td>
</tr>
<tr>
<td>getUtilizationQueries</td>
<td>CodeRef&lt;GetMultilineQueries&gt;</td>
<td>no</td>
<td>Prometheus utilization query.</td>
</tr>
<tr>
<td>humanize</td>
<td>CodeRef&lt;Humanize&gt;</td>
<td>no</td>
<td>Convert Prometheus data to human-readable form.</td>
</tr>
</tbody>
</table>

### 7.5.13. console.cluster-overview/utilization-item

#### 7.5.13.1. Summary

Adds a new cluster overview utilization item.

#### 7.5.13.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>The title of the utilization item.</td>
</tr>
<tr>
<td>getUtilizationQuery</td>
<td>CodeRef&lt;GetQuery&gt;</td>
<td>no</td>
<td>Prometheus utilization query.</td>
</tr>
<tr>
<td>humanize</td>
<td>CodeRef&lt;Humanize&gt;</td>
<td>no</td>
<td>Convert Prometheus data to human-readable form.</td>
</tr>
<tr>
<td>getTotalQuery</td>
<td>CodeRef&lt;GetQuery&gt;</td>
<td>yes</td>
<td>Prometheus total query.</td>
</tr>
<tr>
<td>getRequestQuery</td>
<td>CodeRef&lt;GetQuery&gt;</td>
<td>yes</td>
<td>Prometheus request query.</td>
</tr>
<tr>
<td>getLimitQuery</td>
<td>CodeRef&lt;GetQuery&gt;</td>
<td>yes</td>
<td>Prometheus limit query.</td>
</tr>
</tbody>
</table>
7.5.1.14. console.context-provider

7.5.1.14.1. Summary
Adds a new React context provider to the web console application root.

7.5.1.14.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provider</td>
<td>CodeRef&lt;Provider&lt;T&gt;&gt;</td>
<td>no</td>
<td>Context Provider component.</td>
</tr>
<tr>
<td>useValueHook</td>
<td>CodeRef&lt;() ⇒ T&gt;</td>
<td>no</td>
<td>Hook for the Context value.</td>
</tr>
</tbody>
</table>

7.5.1.15. console.dashboards/card

7.5.1.15.1. Summary
Adds a new dashboard card.

7.5.1.15.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tab</td>
<td>string</td>
<td>no</td>
<td>The ID of the dashboard tab to which the card will be added.</td>
</tr>
<tr>
<td>position</td>
<td>'LEFT'</td>
<td>'RIGHT'</td>
<td>'MAIN'</td>
</tr>
<tr>
<td>component</td>
<td>CodeRef&lt;React.ComponentType&lt;{}&gt;&gt;</td>
<td>no</td>
<td>Dashboard card component.</td>
</tr>
<tr>
<td>span</td>
<td>OverviewCardSpan</td>
<td>yes</td>
<td>Card's vertical span in the column. Ignored for small screens; defaults to 12</td>
</tr>
</tbody>
</table>

7.5.1.16. console.dashboards/overview/activity/resource

7.5.1.16.1. Summary
Adds an activity to the Activity Card of Overview Dashboard where the triggering of activity is based on watching a Kubernetes resource.
### 7.5.16.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>k8sResource</td>
<td><code>CodeRef&lt;FirehoseResource &amp; { isList: true; }&gt;</code></td>
<td>no</td>
<td>The utilization item to be replaced.</td>
</tr>
<tr>
<td>component</td>
<td><code>CodeRef&lt;React.ComponentType&lt;K8sActivityProps&lt;T&gt;&gt;&gt;</code></td>
<td>no</td>
<td>The action component.</td>
</tr>
<tr>
<td>isActivity</td>
<td><code>CodeRef&lt;(resource: T) =&gt; boolean&gt;</code></td>
<td>yes</td>
<td>Function which determines if the given resource represents the action. If not defined, every resource represents activity.</td>
</tr>
<tr>
<td>getTimestamp</td>
<td><code>CodeRef&lt;(resource: T) =&gt; Date&gt;</code></td>
<td>yes</td>
<td>Time stamp for the given action, which will be used for ordering.</td>
</tr>
</tbody>
</table>

### 7.5.17. `console.dashboards/overview/detail/item`

#### 7.5.17.1. Summary

Adds an item to the Details card of Overview dashboard

#### 7.5.17.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td><code>CodeRef&lt;React.ComponentType&lt;{}&gt;&gt;</code></td>
<td>no</td>
<td>The value, based on the DetailItem component</td>
</tr>
</tbody>
</table>

### 7.5.18. `console.dashboards/overview/health/operator`

#### 7.5.18.1. Summary

Adds a health subsystem to the status card of the Overview dashboard, where the source of status is a Kubernetes REST API.

#### 7.5.18.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 7.5.1.19. console.dashboards/overview/health/prometheus

#### 7.5.1.19.1. Summary

Adds a health subsystem to the status card of Overview dashboard where the source of status is Prometheus.

#### 7.5.1.19.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>The display name of the subsystem.</td>
</tr>
<tr>
<td>queries</td>
<td>string[]</td>
<td>no</td>
<td>The Prometheus queries</td>
</tr>
<tr>
<td>healthHandler</td>
<td>CodeRef&lt;PrometheusHealthHandler&gt;</td>
<td>no</td>
<td>Resolve the subsystem’s health.</td>
</tr>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>additionalResource</td>
<td>CodeRef&lt;FirehoseResource&gt;</td>
<td>yes</td>
<td>Additional resource which will be fetched and passed to healthHandler.</td>
</tr>
<tr>
<td>popupComponent</td>
<td>CodeRef&lt;React.ComponentType&lt;PrometheusHealthPopupProps&gt;&gt;</td>
<td>yes</td>
<td>Loader for pop-up menu content. If defined, a health item is represented as a link, which opens a pop-up menu with the given content.</td>
</tr>
<tr>
<td>popupTitle</td>
<td>string</td>
<td>yes</td>
<td>The title of the popover.</td>
</tr>
<tr>
<td>disallowedControlPlaneTopology</td>
<td>string[]</td>
<td>yes</td>
<td>Control plane topology for which the subsystem should be hidden.</td>
</tr>
</tbody>
</table>

### 7.5.1.20. console.dashboards/overview/health/resource

#### 7.5.1.20.1. Summary

Adds a health subsystem to the status card of Overview dashboard where the source of status is a Kubernetes Resource.

#### 7.5.1.20.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>The display name of the subsystem.</td>
</tr>
<tr>
<td>resources</td>
<td>CodeRef&lt;WatchK8sResources&lt;T&gt;&gt;</td>
<td>no</td>
<td>Kubernetes resources that will be fetched and passed to healthHandler.</td>
</tr>
<tr>
<td>healthHandler</td>
<td>CodeRef&lt;ResourceHealthHandler&lt;T&gt;&gt;</td>
<td>no</td>
<td>Resolve the subsystem’s health.</td>
</tr>
<tr>
<td>popupComponent</td>
<td>CodeRef&lt;WatchK8sResults&lt;T&gt;&gt;</td>
<td>yes</td>
<td>Loader for pop-up menu content. If defined, a health item is represented as a link, which opens a pop-up menu with the given content.</td>
</tr>
</tbody>
</table>
7.5.1.21. console.dashboards/overview/health/url

7.5.1.21.1. Summary

Adds a health subsystem to the status card of Overview dashboard where the source of status is a Kubernetes REST API.

7.5.1.21.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>no</td>
<td>The display name of the subsystem.</td>
</tr>
<tr>
<td>url</td>
<td>string</td>
<td>no</td>
<td>The URL to fetch data from. It will be prefixed with base Kubernetes URL.</td>
</tr>
</tbody>
</table>
| healthHandler | `CodeRef<URLHealthHandler<T, K8sResourceCommon>[
|             |            |          | no          |
|             |            |          |             |
| additionalResource | CodeRef<FirehoseResource> | yes | Resolve the subsystem’s health. |
| popupComponent | CodeRef<React.CompomentType<{healthResult?: T; healthResultError?: any; k8sResult?: FirehoseResult<R>}> | yes | Additional resource which will be fetched and passed to healthHandler. |
| popupTitle  | string     | yes      | Loader for popup content. If defined, a health item will be represented as a link which opens popup with given content. |

7.5.1.22. console.dashboards/overview/inventory/item

7.5.1.22.1. Summary
Adds a resource tile to the overview inventory card.

### 7.5.1.22.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>CodeRef&lt;T&gt;</td>
<td>no</td>
<td>The model for resource which will be fetched. Used to get the model's label or abbr.</td>
</tr>
<tr>
<td>mapper</td>
<td>CodeRef&lt;StatusGroupMapper&lt;T, R&gt;&gt;</td>
<td>yes</td>
<td>Function which maps various statuses to groups.</td>
</tr>
<tr>
<td>additionalResources</td>
<td>CodeRef&lt;WatchK8sResources&lt;R&gt;&gt;</td>
<td>yes</td>
<td>Additional resources which will be fetched and passed to the mapper function.</td>
</tr>
</tbody>
</table>

### 7.5.1.23. console.dashboards/overview/inventory/item/group

#### 7.5.1.23.1. Summary

Adds an inventory status group.

#### 7.5.1.23.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>The id of the status group.</td>
</tr>
<tr>
<td>icon</td>
<td>CodeRef&lt;React.ReactElement&lt;any, string</td>
<td>React.JSXElementConstructor&lt;any&gt;&gt;</td>
<td>no</td>
</tr>
</tbody>
</table>

### 7.5.1.24. console.dashboards/overview/inventory/item/replacement

#### 7.5.1.24.1. Summary

Replaces an overview inventory card.

#### 7.5.1.24.2. Properties
<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>model</strong></td>
<td>CodeRef&lt;T&gt;</td>
<td>no</td>
<td>The model for resource which will be fetched. Used to get the model’s label or abbr.</td>
</tr>
<tr>
<td><strong>mapper</strong></td>
<td>CodeRef&lt;StatusGroupMapper&lt;T, R&gt;&gt;</td>
<td>yes</td>
<td>Function which maps various statuses to groups.</td>
</tr>
<tr>
<td><strong>additionalResources</strong></td>
<td>CodeRef&lt;WatchK8sResources&lt;R&gt;&gt;</td>
<td>yes</td>
<td>Additional resources which will be fetched and passed to the mapper function.</td>
</tr>
</tbody>
</table>

7.5.1.25. console.dashboards/overview/prometheus/activity/resource

7.5.1.25.1. Summary

Adds an activity to the Activity Card of Prometheus Overview Dashboard where the triggering of activity is based on watching a Kubernetes resource.

7.5.1.25.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>queries</strong></td>
<td>string[]</td>
<td>no</td>
<td>Queries to watch</td>
</tr>
<tr>
<td><strong>component</strong></td>
<td>CodeRef&lt;React.ComponentType&lt;PrometheusActivityProps&gt;&gt;</td>
<td>no</td>
<td>The action component.</td>
</tr>
<tr>
<td><strong>isActivity</strong></td>
<td>CodeRef&lt;(results: PrometheusResponse[]) ⇒ boolean&gt;</td>
<td>yes</td>
<td>Function which determines if the given resource represents the action. If not defined, every resource represents activity.</td>
</tr>
</tbody>
</table>

7.5.1.26. console.dashboards/project/overview/item

7.5.1.26.1. Summary

Adds a resource tile to the project overview inventory card.

7.5.1.26.2. Properties
### Name | Value Type | Optional | Description
--- | --- | --- | ---
model | `CodeRef<T>` | no | The model for resource which will be fetched. Used to get the model's label or abbr.
mapper | `CodeRef<StatusGroupMapper<T, R>>` | yes | Function which maps various statuses to groups.
additionalResources | `CodeRef<WatchK8sResources<R>>` | yes | Additional resources which will be fetched and passed to the mapper function.

#### 7.5.1.27. console.dashboards/tab

**7.5.1.27.1. Summary**

Adds a new dashboard tab, placed after the Overview tab.

**7.5.1.27.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
</table>
id      | `string`   | no       | A unique tab identifier, used as tab link href and when adding cards to this tab. |

**navSection** | `'home' | 'storage'` | no | Navigation section to which the tab belongs to. |
title    | `string`   | no       | The title of the tab. |

#### 7.5.1.28. console.file-upload

**7.5.1.28.1. Summary**

(not available)

**7.5.1.28.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------</td>
<td>----------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>fileExtensions</td>
<td>string[]</td>
<td>no</td>
<td>Supported file extensions.</td>
</tr>
<tr>
<td>handler</td>
<td>CodeRef&lt;FileUploadHandler&gt;</td>
<td>no</td>
<td>Function which handles the file drop action.</td>
</tr>
</tbody>
</table>

**7.5.1.29. console.flag**

**7.5.1.29.1. Summary**

Gives full control over the web console feature flags.

**7.5.1.29.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>handler</td>
<td>CodeRef&lt;FeatureFlagHandler&gt;</td>
<td>no</td>
<td>Used to set or unset arbitrary feature flags.</td>
</tr>
</tbody>
</table>

**7.5.1.30. console.flag/hookProvider**

**7.5.1.30.1. Summary**

Gives full control over the web console feature flags with hook handlers.

**7.5.1.30.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>handler</td>
<td>CodeRef&lt;FeatureFlagHandler&gt;</td>
<td>no</td>
<td>Used to set or unset arbitrary feature flags.</td>
</tr>
</tbody>
</table>

**7.5.1.31. console.flag/model**

**7.5.1.31.1. Summary**

Adds a new web console feature flag driven by the presence of a CRD on the cluster.

**7.5.1.31.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>flag</td>
<td>string</td>
<td>no</td>
<td>The name of the flag to set once the CRD is detected.</td>
</tr>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>The model which refers to a CustomResourceDefinition.</td>
</tr>
</tbody>
</table>

7.5.1.32. console.global-config

7.5.1.32.1. Summary

(not available)

7.5.1.32.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>Unique identifier for the cluster config resource instance.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>no</td>
<td>The name of the cluster config resource instance.</td>
</tr>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>The model which refers to a cluster config resource.</td>
</tr>
<tr>
<td>namespace</td>
<td>string</td>
<td>no</td>
<td>The namespace of the cluster config resource instance.</td>
</tr>
</tbody>
</table>

7.5.1.33. console.model-metadata

7.5.1.33.1. Summary

Customize the display of models by overriding values retrieved and generated through API discovery.

7.5.1.33.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>model</td>
<td>ExtensionK8sGroup Model</td>
<td>no</td>
<td>The model to customize. May specify only a group, or optional version and kind.</td>
</tr>
<tr>
<td>badge</td>
<td>ModelBadge</td>
<td>yes</td>
<td>Whether to consider this model reference as Technology Preview or Developer Preview.</td>
</tr>
<tr>
<td>color</td>
<td>string</td>
<td>yes</td>
<td>The color to associate to this model.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>yes</td>
<td>Override the label. Requires <strong>kind</strong> be provided.</td>
</tr>
<tr>
<td>labelPlural</td>
<td>string</td>
<td>yes</td>
<td>Override the plural label. Requires <strong>kind</strong> be provided.</td>
</tr>
<tr>
<td>abbr</td>
<td>string</td>
<td>yes</td>
<td>Customize the abbreviation. Defaults to all uppercase characters in <strong>kind</strong>, up to 4 characters long. Requires that <strong>kind</strong> is provided.</td>
</tr>
</tbody>
</table>

7.5.1.34. console.navigation/href

7.5.1.34.1. Summary

(not available)

7.5.1.34.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this item.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>no</td>
<td>The name of this item.</td>
</tr>
<tr>
<td>href</td>
<td>string</td>
<td>no</td>
<td>The link href value.</td>
</tr>
<tr>
<td>Name</td>
<td>Value Type</td>
<td>Optional</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>perspective</td>
<td>string</td>
<td>yes</td>
<td>The perspective ID to which this item belongs to. If not specified, contributes to the default perspective.</td>
</tr>
<tr>
<td>section</td>
<td>string</td>
<td>yes</td>
<td>Navigation section to which this item belongs to. If not specified, render this item as a top level link.</td>
</tr>
<tr>
<td>dataAttributes</td>
<td>{ [key: string]: string; }</td>
<td>yes</td>
<td>Adds data attributes to the DOM.</td>
</tr>
<tr>
<td>startsWith</td>
<td>string[]</td>
<td>yes</td>
<td>Mark this item as active when the URL starts with one of these paths.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>namespaced</td>
<td>boolean</td>
<td>yes</td>
<td>If true, adds /ns/active-namespace to the end.</td>
</tr>
<tr>
<td>prefixNamespaced</td>
<td>boolean</td>
<td>yes</td>
<td>If true, adds /k8s/ns/active-namespace to the beginning.</td>
</tr>
</tbody>
</table>

7.5.1.35. console.navigation/resource-cluster

7.5.1.35.1. Summary

(not available)

7.5.1.35.2. Properties
<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this item.</td>
</tr>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>The model for which this navigation item links to.</td>
</tr>
<tr>
<td>perspective</td>
<td>string</td>
<td>yes</td>
<td>The perspective ID to which this item belongs to. If not specified, contributes to the default perspective.</td>
</tr>
<tr>
<td>section</td>
<td>string</td>
<td>yes</td>
<td>Navigation section to which this item belongs to. If not specified, render this item as a top-level link.</td>
</tr>
<tr>
<td>dataAttributes</td>
<td>{ [key: string]: string; }</td>
<td>yes</td>
<td>Adds data attributes to the DOM.</td>
</tr>
<tr>
<td>startsWith</td>
<td>string[]</td>
<td>yes</td>
<td>Mark this item as active when the URL starts with one of these paths.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>yes</td>
<td>Overrides the default name. If not supplied the name of the link will equal the plural value of the model.</td>
</tr>
</tbody>
</table>

7.5.1.36. `console.navigation/resource-ns`

7.5.1.36.1. Summary

(not available)
### 7.5.1.36.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this item.</td>
</tr>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>The model for which this navigation item links to.</td>
</tr>
<tr>
<td>perspective</td>
<td>string</td>
<td>yes</td>
<td>The perspective ID to which this item belongs to. If not specified, contributes to the default perspective.</td>
</tr>
<tr>
<td>section</td>
<td>string</td>
<td>yes</td>
<td>Navigation section to which this item belongs to. If not specified, render this item as a top-level link.</td>
</tr>
<tr>
<td>dataAttributes</td>
<td>{ [key: string]: string; }</td>
<td>yes</td>
<td>Adds data attributes to the DOM.</td>
</tr>
<tr>
<td>startsWith</td>
<td>string[]</td>
<td>yes</td>
<td>Mark this item as active when the URL starts with one of these paths.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>yes</td>
<td>Overrides the default name. If not supplied the name of the link will equal the plural value of the model.</td>
</tr>
</tbody>
</table>

---

### 7.5.1.37. console.navigation/section

#### 7.5.1.37.1. Summary
7.5.1.37.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this item.</td>
</tr>
<tr>
<td>perspective</td>
<td>string</td>
<td>yes</td>
<td>The perspective ID to which this item belongs to. If not specified, contributes to the default perspective.</td>
</tr>
<tr>
<td>dataAttributes</td>
<td>{ [key: string]: string; }</td>
<td>yes</td>
<td>Adds data attributes to the DOM.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>string[]</td>
<td>yes</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>yes</td>
<td>Name of this section. If not supplied, only a separator will be shown above the section.</td>
</tr>
</tbody>
</table>

7.5.1.38. console.navigation/separator

7.5.1.38.1. Summary

(not available)

7.5.1.38.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this item.</td>
</tr>
</tbody>
</table>
### perspective

**Value Type**: string  
**Optional**: yes  
**Description**: The perspective ID to which this item belongs to. If not specified, contributes to the default perspective.

### section

**Value Type**: string  
**Optional**: yes  
**Description**: Navigation section to which this item belongs to. If not specified, render this item as a top level link.

### dataAttributes

**Value Type**: { [key: string]: string; }  
**Optional**: yes  
**Description**: Adds data attributes to the DOM.

### insertBefore

**Value Type**: string | string[]  
**Optional**: yes  
**Description**: Insert this item before the item referenced here. For arrays, the first one found in order is used.

### insertAfter

**Value Type**: string | string[]  
**Optional**: yes  
**Description**: Insert this item after the item referenced here. For arrays, the first one found in order is used.  
**Note**: `insertBefore` takes precedence.

### 7.5.1.39. console.page/resource/details

#### 7.5.1.39.1. Summary

Adds a new resource details page to the web console router.

#### 7.5.1.39.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ExtensionK8sGroupKindModel</td>
<td>no</td>
<td>The model for which this resource page links to.</td>
</tr>
</tbody>
</table>
### 7.5.1.40. `console.page/resource/list`

#### 7.5.1.40.1. Summary

Adds new resource list page to Console router.

#### 7.5.1.40.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td><code>ExtensionK8sGroupKindModel</code></td>
<td>no</td>
<td>The model for which this resource page links to.</td>
</tr>
<tr>
<td>component</td>
<td><code>CodeRef&lt;React.ComponentType&lt;{match: match&lt;{}&gt;; namespace: string; model: ExtensionK8sModel; }&gt;&gt;</code></td>
<td>no</td>
<td>The component to be rendered when the route matches.</td>
</tr>
</tbody>
</table>

### 7.5.1.41. `console.page/route`

#### 7.5.1.41.1. Summary

Adds a new page to the web console router. See [React Router](#).

#### 7.5.1.41.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td><code>CodeRef&lt;React.ComponentType&lt;RouteComponentProps&lt;{}, StaticContext, any&gt;&gt;&gt;</code></td>
<td>no</td>
<td>The component to be rendered when the route matches.</td>
</tr>
</tbody>
</table>
### Name | Value Type | Optional | Description
---|---|---|---
path | string | no | Valid URL path or array of paths that path-to-regexp@^1.7.0 understands.
perspective | string | yes | The perspective to which this page belongs to. If not specified, contributes to all perspectives.
exact | boolean | yes | When true, will only match if the path matches the location.pathname exactly.

#### 7.5.1.42. console.page/route/standalone

**7.5.1.42.1. Summary**

Adds a new standalone page, rendered outside the common page layout, to the web console router. See React Router.

**7.5.1.42.2. Properties**

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td>CodeRef&lt;React.ComponentType&lt;RouteComponentProps&lt;{}, StaticContext, any&gt;&gt;&gt;</td>
<td>no</td>
<td>The component to be rendered when the route matches.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>no</td>
<td>Valid URL path or array of paths that path-to-regexp@^1.7.0 understands.</td>
</tr>
<tr>
<td>exact</td>
<td>boolean</td>
<td>yes</td>
<td>When true, will only match if the path matches the location.pathname exactly.</td>
</tr>
</tbody>
</table>

#### 7.5.1.43. console.perspective
7.5.1.43.1. Summary

(not available)

7.5.1.43.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>The perspective identifier.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>no</td>
<td>The perspective display name.</td>
</tr>
<tr>
<td>icon</td>
<td>CodeRef&lt;LazyComponent&gt;</td>
<td>no</td>
<td>The perspective display icon.</td>
</tr>
<tr>
<td>landingPageURL</td>
<td>CodeRef&lt;(flags: { [key: string]: boolean; }, isFirstVisit: boolean) ⇒ string&gt;</td>
<td>no</td>
<td>The function to get perspective landing page URL.</td>
</tr>
<tr>
<td>importRedirectURL</td>
<td>CodeRef&lt;(namespace: string) ⇒ string&gt;</td>
<td>no</td>
<td>The function to get redirect URL for import flow.</td>
</tr>
<tr>
<td>default</td>
<td>boolean</td>
<td>yes</td>
<td>Whether the perspective is the default. There can only be one default.</td>
</tr>
<tr>
<td>defaultPins</td>
<td>ExtensionK8sModel[]</td>
<td>yes</td>
<td>Default pinned resources on the nav</td>
</tr>
<tr>
<td>usePerspectiveDetection</td>
<td>CodeRef&lt;() ⇒ [boolean, boolean]&gt;</td>
<td>yes</td>
<td>The hook to detect default perspective</td>
</tr>
</tbody>
</table>

7.5.1.44. console.project-overview/inventory-item

7.5.1.44.1. Summary

Adds a new inventory item into the Project Overview page.

7.5.1.44.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
</table>

57
### 7.5.1.45. `console.project-overview/utilization-item`

#### 7.5.1.45.1. Summary
Adds a new project overview utilization item.

#### 7.5.1.45.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>component</code></td>
<td><code>CodeRef&lt;React.ComponentType&lt;{projectName: string;}&gt;</code></td>
<td>no</td>
<td>The component to be rendered.</td>
</tr>
<tr>
<td><code>title</code></td>
<td><code>string</code></td>
<td>no</td>
<td>The title of the utilization item.</td>
</tr>
<tr>
<td><code>getUtilizationQuery</code></td>
<td><code>CodeRef&lt;GetProjectQuery&gt;</code></td>
<td>no</td>
<td>Prometheus utilization query.</td>
</tr>
<tr>
<td><code>humanize</code></td>
<td><code>CodeRef&lt;Humanize&gt;</code></td>
<td>no</td>
<td>Convert Prometheus data to human-readable form.</td>
</tr>
<tr>
<td><code>getTotalQuery</code></td>
<td><code>CodeRef&lt;GetProjectQuery&gt;</code></td>
<td>yes</td>
<td>Prometheus total query.</td>
</tr>
<tr>
<td><code>getRequestQuery</code></td>
<td><code>CodeRef&lt;GetProjectQuery&gt;</code></td>
<td>yes</td>
<td>Prometheus request query.</td>
</tr>
<tr>
<td><code>getLimitQuery</code></td>
<td><code>CodeRef&lt;GetProjectQuery&gt;</code></td>
<td>yes</td>
<td>Prometheus limit query.</td>
</tr>
<tr>
<td><code>TopConsumerPopover</code></td>
<td><code>CodeRef&lt;React.ComponentType&lt;TopConsumerPopoverProps&gt;&gt;</code></td>
<td>yes</td>
<td>Shows the top consumer popover instead of plain value.</td>
</tr>
</tbody>
</table>

### 7.5.1.46. `console.pvc/alert`

#### 7.5.1.46.1. Summary
(not available)
### 7.5.1.46.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>CodeRef&lt;React.ComponentType&lt;{ pvc: K8sResourceCommon; }&gt;&gt;</td>
<td>no</td>
<td>The alert component.</td>
</tr>
</tbody>
</table>

### 7.5.1.47. console.pvc/create-prop

#### 7.5.1.47.1. Summary

(not available)

#### 7.5.1.47.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>string</td>
<td>no</td>
<td>Label for the create prop action.</td>
</tr>
<tr>
<td>path</td>
<td>string</td>
<td>no</td>
<td>Path for the create prop action.</td>
</tr>
</tbody>
</table>

### 7.5.1.48. console.pvc/delete

#### 7.5.1.48.1. Summary

(not available)

#### 7.5.1.48.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>CodeRef&lt;(pvc: K8sResourceCommon) ⇒ boolean&gt;</td>
<td>no</td>
<td>Predicate that tells whether to use the extension or not.</td>
</tr>
<tr>
<td>onPVCKill</td>
<td>CodeRef&lt;(pvc: K8sResourceCommon) ⇒ Promise&lt;void&gt;&gt;</td>
<td>no</td>
<td>Method for the PVC delete operation.</td>
</tr>
<tr>
<td>alert</td>
<td>CodeRef&lt;React.ComponentType&lt;{ pvc: K8sResourceCommon; }&gt;&gt;</td>
<td>no</td>
<td>Alert component to show additional information.</td>
</tr>
</tbody>
</table>
7.5.1.49. console.pvc/status

7.5.1.49.1. Summary
(not available)

7.5.1.49.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>priority</td>
<td>number</td>
<td>no</td>
<td>Priority for the status component. A larger value means higher priority.</td>
</tr>
<tr>
<td>status</td>
<td>CodeRef&lt;React.ComponentType&lt;{ pvc: K8sResourceComponent; }&gt;&gt;</td>
<td>no</td>
<td>The status component.</td>
</tr>
<tr>
<td>predicate</td>
<td>CodeRef&lt;{pvc: K8sResourceComponent} =&gt; boolean&gt;</td>
<td>no</td>
<td>Predicate that tells whether to render the status component or not.</td>
</tr>
</tbody>
</table>

7.5.1.50. console-redux-reducer

7.5.1.50.1. Summary
Adds new reducer to Console Redux store which operates on plugins.<scope> substate.

7.5.1.50.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>scope</td>
<td>string</td>
<td>no</td>
<td>The key to represent the reducer-managed substate within the Redux state object.</td>
</tr>
<tr>
<td>reducer</td>
<td>CodeRef&lt;Reducer&lt;any, AnyAction&gt;&gt;</td>
<td>no</td>
<td>The reducer function, operating on the reducer-managed substate.</td>
</tr>
</tbody>
</table>

7.5.1.51. console.resource/create

7.5.1.51.1. Summary
7.5.1.51.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>The model for which this create resource page will be rendered.</td>
</tr>
<tr>
<td>component</td>
<td>CodeRef&lt;React.ComponentType&lt;CreateResourceComponentProps&gt;&gt;</td>
<td>no</td>
<td>The component to be rendered when the model matches</td>
</tr>
</tbody>
</table>

7.5.1.52. console.storage-provider

7.5.1.52.1. Summary

(not available)

7.5.1.52.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Component</td>
<td>CodeRef&lt;React.ComponentType&lt;Partial&lt;RouteComponentProps&gt;{{}, StaticContext, any&gt;&gt;&gt;&gt;</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

7.5.1.53. console.tab/horizontalNav

7.5.1.53.1. Summary

(not available)

7.5.1.53.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ExtensionK8sKindVersionModel</td>
<td>no</td>
<td>The model for which this provider show tab.</td>
</tr>
</tbody>
</table>
# 7.5.1.54. console.telemetry/listener

## 7.5.1.54.1. Summary

(not available)

## 7.5.1.54.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>listener</td>
<td>CodeRef&lt;TelemetryEventListener&gt;</td>
<td>no</td>
<td>Listen for telemetry events</td>
</tr>
</tbody>
</table>

# 7.5.1.55. console.topology/adapter/build

## 7.5.1.55.1. Summary

**BuildAdapter** contributes an adapter to adapt element to data that can be used by the **Build** component.

## 7.5.1.55.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapt</td>
<td><code>CodeRef&lt;(element: GraphElement) ⇒ AdapterDataType&lt;Build ConfigData&gt;&gt;</code></td>
<td>undefined</td>
<td>no</td>
</tr>
</tbody>
</table>

# 7.5.1.56. console.topology/adapter/network

## 7.5.1.56.1. Summary
**NetworkAdapter** contributes an adapter to adapt element to data that can be used by the **Networking** component.

### 7.5.1.56.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapt</td>
<td><code>CodeRef&lt;(element: GraphElement) ⇒ NetworkAdapterType undefined&gt;</code></td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.1.57. **console.topology/adapter/pod**

#### 7.5.1.57.1. Summary

**PodAdapter** contributes an adapter to adapt element to data that can be used by the **Pod** component.

#### 7.5.1.57.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapt</td>
<td><code>CodeRef&lt;(element: GraphElement) ⇒ AdapterDataType&lt;Pods AdapterDataType&gt; undefined&gt;</code></td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.1.58. **console.topology/component/factory**

#### 7.5.1.58.1. Summary

Getter for a **ViewComponentFactory**.

#### 7.5.1.58.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFactory</td>
<td><code>CodeRef&lt;ViewComponentFactory&gt;</code></td>
<td>no</td>
<td>Getter for a <strong>ViewComponentFactory</strong></td>
</tr>
</tbody>
</table>

### 7.5.1.59. **console.topology/create/connector**

#### 7.5.1.59.1. Summary

Getter for the create connector function.

#### 7.5.1.59.2. Properties
### 7.5.1.60. console.topology/data/factory

#### 7.5.1.60.1. Summary

Topology Data Model Factory Extension

#### 7.5.1.60.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>Unique ID for the factory.</td>
</tr>
<tr>
<td>priority</td>
<td>number</td>
<td>no</td>
<td>Priority for the factory.</td>
</tr>
<tr>
<td>resources</td>
<td>WatchK8sResourcesGeneric</td>
<td>yes</td>
<td>Resources to be fetched from useK8sWatchResources hook.</td>
</tr>
<tr>
<td>workloadKeys</td>
<td>string[]</td>
<td>yes</td>
<td>Keys in resources containing workloads.</td>
</tr>
<tr>
<td>getDataModel</td>
<td>CodeRef&lt;TopologyDataModelGetter&gt;</td>
<td>yes</td>
<td>Getter for the data model factory.</td>
</tr>
<tr>
<td>isResourceDepicted</td>
<td>CodeRef&lt;TopologyDataModelDepicted&gt;</td>
<td>yes</td>
<td>Getter for function to determine if a resource is depicted by this model factory.</td>
</tr>
<tr>
<td>getDataModelReconciler</td>
<td>CodeRef&lt;TopologyDataModelReconciler&gt;</td>
<td>yes</td>
<td>Getter for function to reconcile data model after all extensions’ models have loaded.</td>
</tr>
</tbody>
</table>

### 7.5.1.61. console.topology/decorator/provider

#### 7.5.1.61.1. Summary

Topology Decorator Provider Extension

#### 7.5.1.61.2. Properties
<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>The ID of this alert. Used to save state if the alert should not be shown after dismissed.</td>
</tr>
<tr>
<td>priority</td>
<td>number</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>quadrant</td>
<td>TopologyQuadrant</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>decorator</td>
<td>CodeRef&lt;TopologyDecoratorGetter&gt;</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

### 7.5.1.62. console.topology/details/resource-alert

#### 7.5.1.62.1. Summary

**DetailsResourceAlert** contributes an alert for specific topology context or graph element.

#### 7.5.1.62.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>The ID of this alert. Used to save state if the alert should not be shown after dismissed.</td>
</tr>
<tr>
<td>contentProvider</td>
<td><code>CodeRef&lt;(element: GraphElement) ⇒ DetailsResourceAlertContent&gt;</code></td>
<td>null</td>
<td>no</td>
</tr>
</tbody>
</table>

### 7.5.1.63. console.topology/details/resource-link

#### 7.5.1.63.1. Summary

**DetailsResourceLink** contributes a link for specific topology context or graph element.

#### 7.5.1.63.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>link</td>
<td><code>CodeRef&lt;(element: GraphElement) ⇒ React.Component&gt;</code></td>
<td>undefined</td>
<td>no</td>
</tr>
</tbody>
</table>
### 7.5.1.64. `console.topology/details/tab`

#### 7.5.1.64.1. Summary

`DetailsTab` contributes a tab for the topology details panel.

#### 7.5.1.64.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>A unique identifier for this details tab.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>no</td>
<td>The tab label to display in the UI.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>yes</td>
<td>Insert this item before the item referenced here. For arrays, the first one found in order is used. The <code>insertBefore</code> value takes precedence.</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>yes</td>
<td>Insert this item after the item referenced here. For arrays, the first one found in order is used. The <code>insertBefore</code> value takes precedence.</td>
</tr>
</tbody>
</table>

### 7.5.1.65. `console.topology/details/tab-section`

#### 7.5.1.65.1. Summary

`DetailsTabSection` contributes a section for a specific tab in the topology details panel.

#### 7.5.1.65.2. Properties
### 7.5.1.66. console.topology/display/filters

#### 7.5.1.66.1. Summary

Topology Display Filters Extension

#### 7.5.1.66.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getTopologyFilters</td>
<td>CodeRef&lt;()&gt; ⇒ TopologyDisplayOption[]</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>applyDisplayOptions</td>
<td>CodeRef&lt;TopologyApplyDisplayOptions&gt;</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>
7.5.1.67. console.topology/relationship/provider

7.5.1.67.1. Summary

Topology relationship provider connector extension

7.5.1.67.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provides</td>
<td>CodeRef&lt;RelationshipProviderProvides&gt;</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>tooltip</td>
<td>string</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>create</td>
<td>CodeRef&lt;RelationshipProviderCreate&gt;</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>priority</td>
<td>number</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

7.5.1.68. console.user-preference/group

7.5.1.68.1. Summary

(not available)

7.5.1.68.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>ID used to identify the user preference group.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>no</td>
<td>The label of the user preference group.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>yes</td>
<td>ID of user preference group before which this group should be placed</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>yes</td>
<td>ID of user preference group after which this group should be placed</td>
</tr>
</tbody>
</table>

7.5.1.69. console.user-preference/item

7.5.1.69.1. Summary
7.5.1.69.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>ID used to identify the user preference item and referenced in insertAfter and insertBefore to define the item order.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>no</td>
<td>The label of the user preference.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>no</td>
<td>The description of the user preference.</td>
</tr>
<tr>
<td>field</td>
<td>UserPreferenceField</td>
<td>no</td>
<td>The input field options used to render the values to set the user preference.</td>
</tr>
<tr>
<td>groupId</td>
<td>string</td>
<td>yes</td>
<td>IDs used to identify the user preference groups the item would belong to.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>yes</td>
<td>ID of user preference item before which this item should be placed.</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>yes</td>
<td>ID of user preference item after which this item should be placed.</td>
</tr>
</tbody>
</table>

7.5.1.70. console.yaml-template

7.5.1.70.1. Summary

YAML templates for editing resources via the yaml editor.

7.5.1.70.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>model</td>
<td>ExtensionK8sModel</td>
<td>no</td>
<td>Model associated with the template.</td>
</tr>
</tbody>
</table>
# 7.5.1.71. dev-console.add/action

## 7.5.1.71.1. Summary

(not available)

## 7.5.1.71.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>ID used to identify the action.</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>no</td>
<td>The label of the action.</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>no</td>
<td>The description of the action.</td>
</tr>
<tr>
<td>href</td>
<td>string</td>
<td>no</td>
<td>The href to navigate to.</td>
</tr>
<tr>
<td>groupId</td>
<td>string</td>
<td>yes</td>
<td>IDs used to identify the action groups the action would belong to.</td>
</tr>
<tr>
<td>icon</td>
<td>CodeRef&lt;React.ReactNode&gt;</td>
<td>yes</td>
<td>The perspective display icon.</td>
</tr>
<tr>
<td>accessReview</td>
<td>AccessReviewResourceAttributes[]</td>
<td>yes</td>
<td>Optional access review to control the visibility or enablement of the action.</td>
</tr>
</tbody>
</table>

# 7.5.1.72. dev-console.add/action-group

## 7.5.1.72.1. Summary

(not available)
7.5.1.72.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>string</td>
<td>no</td>
<td>ID used to identify the action group.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>no</td>
<td>The title of the action group.</td>
</tr>
<tr>
<td>insertBefore</td>
<td>string</td>
<td>yes</td>
<td>ID of action group before which this group should be placed</td>
</tr>
<tr>
<td>insertAfter</td>
<td>string</td>
<td>yes</td>
<td>ID of action group after which this group should be placed</td>
</tr>
</tbody>
</table>

7.5.1.73. dev-console.import/environment

7.5.1.73.1. Summary

(not available)

7.5.1.73.2. Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Value Type</th>
<th>Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageStreamName</td>
<td>string</td>
<td>no</td>
<td>Name of the image stream to provide custom environment variables for</td>
</tr>
<tr>
<td>imageStreamTags</td>
<td>string[]</td>
<td>no</td>
<td>List of supported image stream tags</td>
</tr>
<tr>
<td>environments</td>
<td>ImageEnvironment[]</td>
<td>no</td>
<td>List of environment variables</td>
</tr>
</tbody>
</table>

7.5.1.74. console.page/resource/tab

7.5.1.74.1. Summary [DEPRECATED]

Deprecated. Use console.tab/horizontalNav instead. Adds a new resource tab page to Console router.

7.5.1.74.2. Properties
7.5.2. Adding a tab to the pods page

The following procedure adds a tab to the Pod Details page as an example extension to your plug-in.

Procedure

1. Add the following to the console-extensions.json file:

   ```json
   {
     "type": "console.tab/horizontalNav",
     "properties": {
       "page": {
         "name": "Example Tab",
         "href": "example"
       },
       "model": {
         "group": "core",
         "version": "v1",
         "kind": "Pod"
       },
       "component": {
         "$codeRef": "ExampleTab"
       }
     }
   }
   ```

2. Edit the package.json file to include the following changes:

   ```json
   "exposedModules": {
   "ExamplePage": ".components/ExamplePage",
   "ExampleTab": ".components/ExampleTab"
   }
   ```
3. Write a message to display on a new custom tab on the Pods page by creating a new file `src/components/ExampleTab.tsx` and adding the following script:

```javascript
import * as React from 'react';

export default function ExampleTab() {
  return (
    <p>This is a custom tab added to a resource using a dynamic plug-in.</p>
  );
}
```

**Verification**

- Visit a Pod page to view the added tab.

### 7.6. BUILD AN IMAGE WITH DOCKER

To deploy your plug-in on a cluster, you need to build an image and push it to an image registry.

**Procedure**

1. Build the image with the following command:

   ```bash
   $ docker build -t quay.io/my-repository/my-plugin:latest .
   ```

2. Optional: If you want to test your image, run the following command:

   ```bash
   $ docker run -it --rm -d -p 9001:80 quay.io/my-repository/my-plugin:latest
   ```

3. Push the image by running the following command:

   ```bash
   $ docker push quay.io/my-repository/my-plugin:latest
   ```

### 7.7. DEPLOY YOUR PLUG-IN ON A CLUSTER

After pushing an image with your changes to a registry, you can deploy the plug-in to a cluster.

**Procedure**

1. To deploy your plug-in to a cluster, instantiate your plug-in by running the following command:

   ```bash
   $ oc process -f template.yaml \
   -p PLUGIN_NAME=my-plugin \ 1
   -p NAMESPACE=my-plugin-namespace \ 2
   -p IMAGE=quay.io/my-repository/my-plugin:latest \ 3
   | oc create -f -
   ```

   1. Update with the name of your plug-in.
2. Update with the namespace.

3. Update with the name of the image you created.

This command runs a light-weight NGINX HTTP server to serve the assets for your plug-in.

**IMPORTANT**

**PLUGIN_NAME** must match the plug-in name you used in the `consolePlugin` declaration of `package.json`.

2. Patch the Console Operator configuration to enable the plug-in by running the following command:

```
$ oc patch consoles.operator.openshift.io cluster --patch '{ "spec": { "plugins": ["my-plugin"] } }' --type=merge
```

### 7.8. DISABLING YOUR PLUG-IN IN THE BROWSER

Console users can use the `disable-plugins` query parameter to disable specific or all dynamic plug-ins that would normally get loaded at run-time.

**Procedure**

- To disable a specific plug-in(s), remove the plug-in you want to disable from the comma-separated list of plug-in names.

- To disable all plug-ins, leave an empty string in the `disable-plugins` query parameter.

**NOTE**

Cluster administrators can disable plug-ins in the Cluster Settings page of the web console.
You can launch an embedded command line terminal instance in the 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.

### 8.1. 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 fast 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.

**NOTE**

Prior to OpenShift Container Platform 4.8, the Web Terminal Operator bundled all its functionality in a single Operator. As of OpenShift Container Platform 4.8, the Web Terminal Operator installs the DevWorkspace Operator as a dependency to provide the same features.

7. After the Operator is installed, refresh your page to see the command line terminal icon on the upper right of the console.

### 8.2. 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 ( ![Command Line Terminal](image.png) ) 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`, `kubectl`, `odo`, `kn`, `tkn`, `helm`, `kubens`, `subctl`, and `kubectx` in the web terminal.

4. Click `+` to open multiple tabs within web terminal in the console.

**NOTE**

- You can re-run commands by selecting them from the list of commands you have run in the terminal. These commands persist across multiple terminal sessions.
- The web terminal remains open until you close it or you close the browser window or tab.

### 8.3. UNINSTALLING THE WEB TERMINAL

Uninstalling the web terminal is a two-step process:

1. Uninstall the Web Terminal Operator and related custom resources (CRs) that were added when you installed the Operator.
2. Uninstall the DevWorkspace Operator and its related custom resources that were added as a dependency of 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.

8.3.1. Removing the Web Terminal Operator and the custom resources that support it

Use the console and the CLI to delete any existing web terminals and CRs that were created during the installation of the Web Terminal Operator.

**NOTE**

Prior to OpenShift Container Platform 4.8, the Web Terminal Operator used different CRDs to provide Web Terminal capabilities. To uninstall versions 1.2.1 and below of the Web Terminal Operator, refer to the documentation for OpenShift Container Platform 4.7.

Procedure

1. Uninstall the Web Terminal Operator using the web console:
   a. In the Administrator perspective of the web console, navigate to **Operators → Installed Operators**.
   b. Scroll the filter list or type a keyword into the **Filter by name** box to find the **Web Terminal Operator**.
   c. Click the Options menu for the Web Terminal Operator, and then select **Uninstall Operator**.
   d. 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.

2. Remove the CRs used by the Operator.

```shell
$ oc delete devworkspaces.workspace.devfile.io --all-namespaces \--selector 'console.openshift.io/terminal=true' --wait

$ oc delete devworkspacetemplates.workspace.devfile.io --all-namespaces \--selector 'console.openshift.io/terminal=true' --wait
```

8.3.2. Deleting the DevWorkspace Operator dependency
Use the CLI to delete the custom resource definitions (CRDs) and additional resources that are created during installation of the Web Terminal Operator.

**IMPORTANT**

The DevWorkspace Operator functions as a standalone Operator and may be required as a dependency for other Operators installed on the cluster (for example, the Red Hat OpenShift Dev Spaces Operator may depend on it). Follow the steps below only if you are sure the DevWorkspace Operator is no longer needed.

**Procedure**

1. Remove the DevWorkspace custom resources used by the Operator, along with any related Kubernetes objects, such as deployments.

   ```bash
   $ oc delete devworkspaces.workspace.devfile.io --all-namespaces --all --wait
   $ oc delete devworkspaceroutings.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. Remove the CRDs used by the Operator:

   ```bash
   $ oc delete customresourcedefinitions.apiextensions.k8s.io devworkspaceroutings.controller.devfile.io
   $ oc delete customresourcedefinitions.apiextensions.k8s.io devworkspaces.workspace.devfile.io
   $ oc delete customresourcedefinitions.apiextensions.k8s.io devworkspacetemplates.workspace.devfile.io
   ```

   **WARNING**

   The DevWorkspace Operator provides custom resource definitions (CRDs) that use conversion webhooks. Failing to remove these CRDs can cause issues on the cluster.
3. Verify that all involved custom resource definitions are removed. The following command should not display any result.

```bash
$ oc get customresourcedefinitions.apiextensions.k8s.io | grep "devfile.io"
```

4. Remove the `devworkspace-webhook-server` deployment, mutating, and validating webhooks:

```bash
$ oc delete deployment/devworkspace-webhook-server -n openshift-operators
$ oc delete mutatingwebhookconfigurations controller.devfile.io
$ oc delete validatingwebhookconfigurations controller.devfile.io
```

**NOTE**

If you remove the `devworkspace-webhook-server` deployment without removing the mutating and validating webhooks, you will not be able to use `oc exec` commands to run commands in a container on the cluster. After you remove the webhooks you will be able to use the `oc exec` commands again.

5. Remove any remaining services, secrets, and config maps. Depending on the installation, some resources included in the following command may not exist on the cluster.

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

6. Uninstall the Operator using the web console:

   a. In the **Administrator** perspective of the web console, navigate to **Operators → Installed Operators**.

   b. Scroll the filter list or type a keyword into the **Filter by name** box to find the **DevWorkspace** Operator.

   c. Click the Options menu for the DevWorkspace Operator, and then select **Uninstall Operator**.
d. 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 9. DISABLING THE WEB CONSOLE IN OPENSHIFT CONTAINER PLATFORM

You can disable the OpenShift Container Platform web console.

9.1. PREREQUISITES

- Deploy an OpenShift Container Platform cluster.

9.2. 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:

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

  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 10. 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.

10.1. 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

10.2. 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.

### 10.3. 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**

### 10.4. 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.

10.4.1. 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.

10.4.2. 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.

10.4.2.1. conclusion element

Viewing the conclusion element in the YAML file

```yaml
...  
  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.
10.4.2.2. description element

Viewing the description element in the YAML file

```yaml
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.'
```

10.4.2.2. description element

Viewing the description element in the web console

The description appears on the introductory tile of the quick start on the Quick Starts page.
10.4.2.3. displayName element

Viewing the displayName element in the YAML file

```yaml
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.'
  display_name: Get started with Spring
  duration_minutes: 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.
10.4.2.4. durationMinutes element

Viewing the durationMinutes element in the YAML file

```yaml
apiVersion: console.openshift.io/v1
classification: 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
```

The `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.
10.4.2.5. icon element

Viewing the icon element in the YAML file

```yaml
spec:
description: 'Import a Spring Application from git, build, and deploy it onto OpenShift.'
displayName: Get started with Spring
durationMinutes: 10
icon: |
  data:image/svg+xml;base64,PHN2ZyB4bWxucz0iaHR0cDovL3d3dy5mcm9ucy5ydS8=
  PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdmciIGlkPSJlZmxhdGUiIHZpZXdCb3g9IjEwMCUiIjByPSJ6cmFkaWduIiB3aWR0aD0iMTAiIGhlaWdodD0iMTAiIHZpb3Q9IiBTMEUiIGRyYW5zIjogIiBkYXRhLWZhbWlvdW50YWxhenk9IiBTMEUiIGRyYW5zIjogIiBkYXRhLWZhbWlvdW50YWxheW91ciIgUGxhY2h5PSIjMzg3OSI+
```

OpenShift Container Platform 4.11 Web console
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.
Viewing the introduction element in the YAML file

```
... introduction: >-
  **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
```

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.
**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
10.4.3. Adding a custom icon to a quick start

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

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,PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdmciIHJvbGU9ImltZyIgdmlld.
```

10.4.4. 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
```

10.4.5. 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
```
10.4.6. Supported tags for quick starts

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

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

10.4.7. 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

### 10.4.7.1. Perspective switcher

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

### 10.4.7.2. 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}}
```

### 10.4.7.3. 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}}
```

### 10.4.7.4. Common navigation links

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

### 10.4.7.5. 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}}
```

### 10.4.8. Code snippet markdown reference
You can execute a CLI code snippet when it is included in a quick start from the web console. To use this feature, you must first install the Web Terminal Operator. The web terminal and code snippet actions that execute in the web terminal are not present if you do not install the Web Terminal Operator. Alternatively, you can copy a code snippet to the clipboard regardless of whether you have the Web Terminal Operator installed or not.

10.4.8.1. Syntax for inline code snippets
```
'code block'{{copy}}
'code block'{{execute}}
```

**NOTE**
If the `execute` syntax is used, the Copy to clipboard action is present whether you have the Web Terminal Operator installed or not.

10.4.8.2. Syntax for multi-line code snippets
```
```
```
multi line code block
```{{copy}}
```
```
multi line code block
```
```
multi line code block
```{{execute}}
```

10.5. QUICK START CONTENT GUIDELINES

10.5.1. 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.
```

10.5.2. 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

### 10.5.3. 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.

10.5.4. 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.

10.5.5. 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.

10.6. 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](#).