OpenShift Container Platform 4.5

Web コンソール

OpenShift Container Platform Web コンソールのスタートガイド

Last Updated: 2021-07-13

OpenShift Container Platform 4.5 Web コンソール

OpenShift Container Platform Web コンソールのスタートガイド

法律上の通知

概要

本書では、OpenShift Container Platform Web コンソールにアクセスし、カスタマイズする方法を説明します。

目次

第1章 WEB コンソールへのアクセス

OpenShift Container Platform Web コンソールは、Web ブラウザーからアクセスできるユーザーインターフェースです。開発者は Web コンソールを使用してプロジェクトのコンテンツを視覚的に把握し、参照し、管理することができます。

1.1. 前提条件

- Web コンソールを使用するために JavaScript が有効にされている必要があります。WebSocket をサポートする Web ブラウザーを使用することが最も推奨されます。
- 「OpenShift Container Platform 4.x のテスト済みインテグレーション」のページを確認してから、クラスターのサポートされるインフラストラクチャーを作成します。

1.2. WEB コンソールの理解および WEB コンソールへのアクセス

Web コンソールはマスター上で Pod として実行されます。Web コンソールを実行するために必要な静的アセットは Pod によって提供されます。OpenShift Container Platform が **openshift-install create cluster** を使用して正常にインストールされた後に、Web コンソールの URL およびインストールされたクラスターのログイン認証情報を、インストールプログラムの CLI 出力で確認します。以下は例になります。

出力例

```
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>
```

これらの詳細を使用してログインし、Web コンソールにアクセスします。

インストールしていない既存のクラスターの場合、**oc whoami --show-console** を使用して Web コンソール URL を表示します。

第2章 OPENSHIFT CONTAINER PLATFORM DASHBOARD を使用したクラスター情報の取得

OpenShift Container Platform Web コンソールから **Home → Dashboards → Overview** に移動し、クラスターの概要情報をキャプチャーする OpenShift Container Platform ダッシュボードにアクセスします。

OpenShift Container Platform ダッシュボードは、個別のダッシュボードカードでキャプチャーされるさまざまなクラスター情報を提供します。

2.1. OPENSHIFT CONTAINER PLATFORM ダッシュボードページについて

OpenShift Container Platform ダッシュボードは以下のカードで構成されます。

- **Details** は、クラスターの詳細情報の概要を表示します。
  ステータスには、**ok**、**error**、**warning**、**in progress**、および **unknown** が含まれます。リソースでは、カスタムのステータス名を追加できます。
  - クラスター
  - プロバイダー
  - バージョン
- **Cluster Inventory** は、リソースの数および関連付けられたステータスの詳細を表示します。これは、問題の解決に介入が必要な場合に役立ちます。以下についての情報が含まれます。
  - ノード数
  - Pod 数
  - 永続ストレージボリューム要求
  - クラスター内のベアメタルホスト。これらはステータス別に一覧表示されます (**metal3** 環境でのみ利用可能です)。
- **Cluster Capacity** チャートは、管理者が追加リソースがクラスターで必要になるタイミングを把握するのに役立ちます。このチャートには、現在の消費量を表示する内側の円が含まれ、外側の円には、以下の情報を含む、リソースに対して設定されたしきい値が表示されます。
  - CPU 時間
  - メモリー割り当て
  - 消費されるストレージ
  - 消費されるネットワークリソース
- **Cluster Utilization** は指定された期間における各種リソースの容量を表示します。これは、管理者がリソースの高い消費量の規模および頻度を理解するのに役立ちます。
- **Events** は、Pod の作成または別のホストへの仮想マシンの移行などのクラスター内の最近のアクティビティーに関連したメッセージを一覧表示します。

- **Top Consumers** は、管理者がクラスターリソースが消費される状況を把握するのに役立ちます。リソースをクリックし、指定されたクラスターリソース (CPU、メモリー、またはストレージ) の最大量を消費する Pod およびノードを一覧表示する詳細ページに切り替えます。

第3章 OPENSHIFT CONTAINER PLATFORM の WEB コンソールの設定

OpenShift Container Platform の Web コンソールを変更してログアウトリダイレクト URL を設定したり、コンソールを無効にしたりすることができます。

3.1. 前提条件

- OpenShift Container Platform クラスターをデプロイします。

3.2. WEB コンソールの設定

console.config.openshift.io リソースを編集して Web コンソールを設定できます。

- **console.config.openshift.io** リソースを編集します。

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

以下の例は、コンソールのリソース定義のサンプルを示しています。

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

(1) ユーザーが Web コンソールからログアウトする際に読み込むページの URL を指定します。値を指定しない場合、ユーザーは Web コンソールのログインページに戻ります。**logoutRedirect** URL を指定することにより、ユーザーはアイデンティティープロバイダー経由でシングルログアウト (SLO) を実行し、シングルサインオンセッションを破棄することができます。

(2) Web コンソール URL。これをカスタム値に更新するには、「**Web コンソール URL のカスタマイズ**」を参照してください。

第4章 OPENSHIFT CONTAINER PLATFORM の WEB コンソールのカスタマイズ

OpenShift Container Platform の Web コンソールをカスタマイズして、カスタムロゴ、製品名、リンク、通知、およびコマンドラインのダウンロードを設定できます。これは、Web コンソールを企業や政府の特定要件を満たすように調整する必要がある場合にとくに役立ちます。

4.1. カスタムロゴおよび製品名の追加

カスタムロゴまたはカスタム製品名を追加することで、カスタムブランディングを作成できます。これらの設定は相互に独立しているため、両方またはいずれかを設定できます。

前提条件

- 管理者の権限があること。
- 使用するロゴのファイルを作成します。ロゴは、GIF、JPG、PNG、または SVG を含む共通のイメージ形式のファイルであり、**60px** の **max-height** に制限されます。

手順

1. ロゴファイルを **openshift-config** namespace の設定マップにインポートします。

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

2. Web コンソールの Operator 設定を編集して、 **customLogoFile** および **customProductName** を組み込みます。

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

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

Operator 設定が更新されると、カスタムロゴ設定マップをコンソール namespace に同期し、これをコンソール Pod にマウントし、再デプロイします。

3. 正常に実行されたかどうかを確認します。問題がある場合は、コンソールクラスター Operator は **Degraded** ステータスを報告し、コンソール Operator 設定も **CustomLogoDegraded** ステータスを **KeyOrFilenameInvalid** または**NoImageProvided** などの理由と共に報告します。**clusteroperator** を確認するには、以下を実行します。

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

コンソール Operator 設定を確認するには、以下を実行します。

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

4.2. WEB コンソールでのカスタムリンクの作成

前提条件

- 管理者の権限があること。

手順

1. **Administration → Custom Resource Definitions**から、 **ConsoleLink** をクリックします。
2. **Instances** タブを選択します。
3. **Create Console Link** をクリックし、ファイルを編集します。

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

1 有効な場所の設定は、**HelpMenu**、**UserMenu**、**ApplicationMenu**、および **NamespaceDashboard** です。

カスタムリンクがすべての namespace に表示されるようにするには、以下の例に従います。

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

カスタムリンクが一部の namespace のみに表示されるようにするには、以下の例に従います。

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

```
    # for these specific namespaces
    - my-namespace
    - your-namespace
    - other-namespace
```

カスタムリンクがアプリケーションメニューに表示されるようにするには、以下の例に従います。

```
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. **Save** ボタンをクリックして変更を適用します。

4.3. WEB コンソール URL のカスタマイズ

Web コンソール URL **consoleURL** をカスタム値に更新できます。

手順

1. **consoles.operator.openshift.io** カスタムリソースでのインストール時にデフォルトで作成されるクラスターインスタンスを変更します。

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

2. カスタム証明書を指定する場合、キーおよび証明書を持つシークレットを **openshift-config** namespace に作成する必要があります。以下は例になります。

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

次に、設定リソースに以下のスタンザを追加します。

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

4.4. ログインページのカスタマイズ

サービス利用規約情報をカスタムログインページを使用して作成します。カスタムログインページは、GitHub や Google などのサードパーティーログインプロバイダーを使用している場合にも、ユーザーが信頼し、予想できるブランドのページを提示して、その後にユーザーを認証プロバイダーにリダイレ

クトする際に役立ちます。また、認証プロセス中にカスタムエラーページをレンダリングすることもできます。

注記

エラーテンプレートのカスタマイズは、要求ヘッダーや OIDC ベースの IDP などのリダイレクトを使用するアイデンティティープロバイダー (IDP) に限定されます。LDAP や HTPasswd などのダイレクトパスワード認証を使用する IDP にはこれによる影響がありません。

前提条件

- 管理者の権限があること。

手順

1. 以下のコマンドを実行して、変更可能なテンプレートを作成します。

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

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

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

2. シークレットを作成します。

```
$ 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. 以下を実行します。

```
$ oc edit oauths cluster
```

4. 仕様を更新します。

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

oc explain oauths.spec.templates を実行して、オプションを把握します。

4.5. 外部ログリンクのテンプレートの定義

ログの参照に役立つサービスに接続しているものの、特定の方法で URL を生成する必要がある場合は、リンクのテンプレートを定義できます。

前提条件

- 管理者の権限があること。

手順

1. **Administration → Custom Resource Definitions**から、 **ConsoleExternalLogLink** をクリックします。
2. **Instances** タブを選択します。
3. **Create Console External Log Link**をクリックし、ファイルを編集します。

```
apiVersion: console.openshift.io/v1
kind: ConsoleExternalLogLink
metadata:
  name: example
spec:
  hrefTemplate: >-
    https://example.com/logs?
resourceName=${resourceName}&containerName=${containerName}&resourceNamespace=$
{resourceNamespace}&podLabels=${podLabels}
  text: Example Logs
```

4.6. カスタム通知バナーの作成

前提条件

- 管理者の権限があること。

手順

1. **Administration → Custom Resource Definitions**から、 **ConsoleNotification** をクリックします。
2. **Instances** タブを選択します。
3. **Create Console Notification** をクリックし、ファイルを編集します。

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

1 有効な場所の設定は、**BannerTop**、**BannerBottom**、および **BannerTopBottom** です。

4. **Create** ボタンをクリックし、変更を適用します。

4.7. CLI ダウンロードのカスタマイズ

ファイルパッケージを直接ポイントしたり、パッケージを提供する外部ページをポイントできるカスタムのリンクテキストおよび URL を使用して、CLI をダウンロードするリンクを設定できます。

前提条件

- 管理者の権限があること。

手順

1. **Administration** → **Custom Resource Definitions**に移動します。
2. カスタムリソース定義 (CRD) の一覧から **ConsoleCLIDownload** を選択します。
3. **YAML** タブをクリックし、編集を行います。

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

4. **Save** ボタンをクリックします。

4.8. YAML サンプルの KUBERNETES リソースへの追加

YAML サンプルはいつでも Kubernetes リソースに動的に追加できます。

前提条件

- クラスター管理者の権限があること。

手順

1. **Administration** → **Custom Resource Definitions**から、**ConsoleYAMLSample** をクリックします。
2. **YAML** をクリックし、ファイルを編集します。

```
apiVersion: console.openshift.io/v1
kind: ConsoleYAMLSample
metadata:
  name: example
spec:
  targetResource:
    apiVersion: batch/v1
    kind: Job
  title: Example Job
  description: An example Job YAML sample
  yaml: |
    apiVersion: batch/v1
    kind: Job
    metadata:
      name: countdown
    spec:
      template:
        metadata:
          name: countdown
        spec:
          containers:
          - name: counter
            image: centos:7
            command:
            - "bin/bash"
            - "-c"
            - "for i in 9 8 7 6 5 4 3 2 1 ; do echo $i ; done"
          restartPolicy: Never
```

spec.snippet を使用して、YAML サンプルが完全な YAML リソース定義ではなく、ユーザーのカーソルで既存の YAML ドキュメントに挿入できる断片を示します。

3. **Save** をクリックします。

第5章 WEB コンソールの DEVELOPER パースペクティブ

OpenShift Container Platform Web コンソールは、**Administrator** パースペクティブと **Developer** パースペクティブという 2 つのパースペクティブを提供します。

Developer パースペクティブは、以下を実行する機能を含む、開発者のユースケースに固有のワークフローを提供します。

- 既存のコードベース、イメージ、および Dockerfile をインポートして、OpenShift Container Platform でアプリケーションを作成し、デプロイします。
- アプリケーション、コンポーネント、およびプロジェクト内のこれらに関連付けられたサービスと視覚的に対話し、それらのデプロイとビルドの状態をモニターします。
- アプリケーション内のコンポーネントを分類し、コンポーネントの接続をアプリケーション内で、また複数のアプリケーションにまたがって実行します。
- Serverless 機能 (テクノロジープレビュー) を統合します。
- Eclipse Che を使用してアプリケーションコードを編集するためのワークスペースを作成します。

5.1. 前提条件

Developer パースペクティブにアクセスするために、Web コンソールにログインしていること。

5.2. DEVELOPER パースペクティブへのアクセス

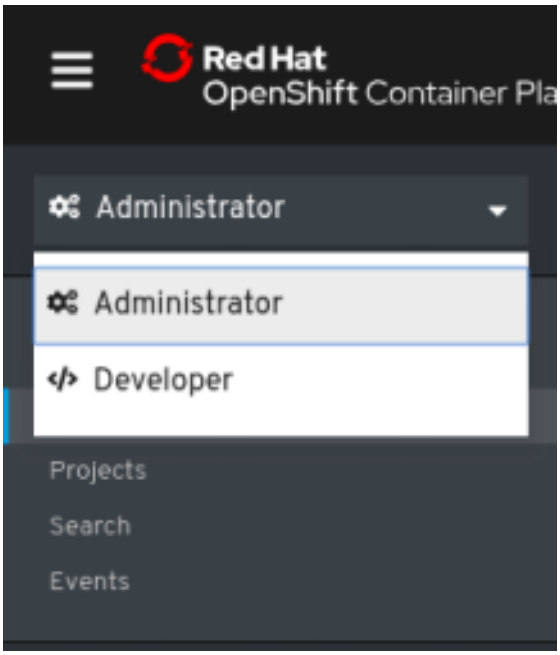
OpenShift Container Platform Web コンソールの **Developer** パースペクティブは、開発者のユースケースに固有のワークフローを提供します。

以下のように、Web コンソールから **Developer** パースペクティブにアクセスできます。

手順

1. ログイン認証情報を使用して OpenShift Container Platform Web コンソールにログインします。OpenShift Container Platform Web コンソールのデフォルトビューは **Administrator** パースペクティブです。
2. パースペクティブスイッチャーを使用して、**Developer** パースペクティブに切り替えます。**Topology** ビューがクラスター内のすべてのプロジェクトの一覧と共に表示されます。

図5.1 Developer パースペクティブ



3. 一覧から既存プロジェクトを選択するか、または **Project** ドロップダウンリストを使用して新規プロジェクトを作成します。

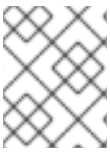
プロジェクト内にワークロードやアプリケーションがない場合、**Topology** ビューにはアプリケーションを作成するための利用可能なオプションが表示されます。既存のワークロードがある場合、**Topology** ビューはワークロードノードをグラフィカルに表示します。

追加リソース

- **Developer** パースペクティブを使用して OpenShift Container Platform でアプリケーションを作成し、デプロイする
- **Topology** ビューを使用してプロジェクトにアプリケーションを表示し、デプロイメントのステータスを確認し、それらと対話する

第6章 WEB コンソールの WEB 端末について

OpenShift Web コンソールで組み込みコマンドラインターミナルインスタンスを起動できます。Web 端末を使用するには、まず Web 端末 Operator をインストールする必要があります。



注記

クラスター管理者は、OpenShift Container Platform 4.7 以降の Web 端末にアクセスできます。

この端末のインスタンスは、**oc**、**kubectl**、**odo**、**kn**、**tkn**、**helm**、**kubens**、および **kubectx** などのクラスターと対話するための一般的な CLI ツールと共に事前にインストールされます。また、これには作業しているプロジェクトのコンテキストが含まれ、ユーザーの認証情報を使用してユーザーのログインを自動的に行います。

重要

Web 端末はテクノロジープレビュー機能としてのみご利用いただけます。テクノロジープレビュー機能は Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。Red Hat は実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

Red Hat のテクノロジープレビュー機能のサポート範囲についての詳細は、https://access.redhat.com/ja/support/offerings/techpreview/ を参照してください。

6.1. WEB 端末のインストール

OpenShift Container Platform OperatorHub に一覧表示されている Operator を使用して Web 端末をインストールできます。Web 端末 Operator をインストールする際に、**DevWorkspace** CRD などのコマンドラインの設定に必要なカスタムリソース定義 (CRD) が自動的にインストールされます。Web コンソールでは、Web 端末を開く際に必要なリソースを作成します。

前提条件

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。

手順

1. Web コンソールの **Administrator** パースペクティブで、**Operators → OperatorHub** に移動します。
2. **Filter by keyword** ボックスを使用してカタログで **Web Terminal** Operator を検索し、**Web Terminal** タイルをクリックします。
3. **Web Terminal** ページで Operator についての簡単な説明を確認してから、**Install** をクリックします。
4. **Install Operator** ページで、すべてのフィールドのデフォルト値を保持します。

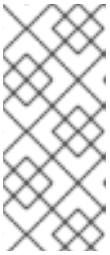
- **Update Channel** メニューの **alpha** オプションは、Web 端末 Operator の最新リリースのインストールを可能にします。
- **Installation Mode** メニューの **All namespaces on the cluster**オプションにより、Operator にクラスターのすべての namespace を監視され、Operator をこれらの namespace で利用可能にすることができます。
- **Installed Namespace** メニューの **openshift-operators** オプションは、Operator をデフォルトの **openshift-operators** namespace にインストールします。
- **Approval Strategy** メニューの **Automatic** オプションにより、Operator への今後のアップグレードは Operator Lifecycle Manager によって自動的に処理されます。

5. **Install** をクリックします。
6. **Installed Operators** ページで、**View operator** をクリックし、Operator が **Installed Operators** ページに一覧表示されていることを確認します。
7. Operator のインストール後に、ページを更新し、コンソールの右上にあるコマンドラインターミナルアイコンを確認します。

6.2. WEB 端末の使用

Web 端末 Operator のインストール後に、以下のように Web 端末を使用できます。

1. Web 端末を起動するには、コンソールの右上にあるコマンドラインターミナルアイコン () をクリックします。Web 端末インスタンスが、**Command line terminal** ペインに表示されます。このインスタンスは、お使いの認証情報を使用して自動的にログインします。
2. **Project** ドロップダウンリストから **DevWorkspace** CR を作成する必要のあるプロジェクトを選択します。デフォルトでは、現在のプロジェクトが選択されます。



注記

- **DevWorkspace** CR は存在しない場合にのみ作成されます。
- **openshift-terminal** プロジェクトは、クラスター管理者に使用されるデフォルトのプロジェクトです。別のプロジェクトを選択するオプションはありません。

3. **Start** をクリックし、選択したプロジェクトを使用して Web 端末を初期化します。

Web 端末を初期化した後に、Web 端末で **oc**、**kubectl**、**odo**、**kn**、**tkn**、**helm**、**kubens** および **kubectx** などの事前インストールされた CLI ツールを使用できます。

6.3. WEB 端末のアンインストール

Web 端末のアンインストールは 2 つの手順で実行されます。

1. Operator のインストール時に追加されたコンポーネントおよびカスタムリソース (CR) を削除します。
2. Web 端末 Operator をアンインストールします。

Web 端末 Operator をアンインストールしても、Operator のインストール時に作成されるカスタムリソース定義 (CRD) または管理リソースは削除されません。これらのコンポーネントは、セキュリティー上の目的で手動でアンインストールする必要があります。これらのコンポーネントを削除すると、Operator のアンインストール時に端末がアイドル状態にならないようにしてクラスターリソースを保存することもできます。

前提条件

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。

6.3.1. Web 端末コンポーネントおよびカスタムリソースの削除

CLI を使用して、Web 端末 Operator のインストール時に作成された CR を削除します。

手順

1. 以下のコマンドを実行して、すべての **DevWorkspace** CR がデプロイメントなどの関連する Kubernetes オブジェクトと共に削除されるようにします。

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

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

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

警告

この手順が完了しない場合、ファイナライザーは Operator を簡単に、かつ完全にアンインストールすることができないようにします。

2. CRD を削除するには、以下のコマンドを実行します。

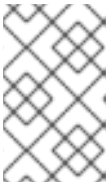
```
$ oc delete customresourcedefinitions.apiextensions.k8s.io
workspaceroutings.controller.devfile.io
```

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

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

3. **DevWorkspace-Webhook-Server** デプロイメントを削除します。

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



注記

これと以下の手順を実行する際に、**oc exec** コマンドを使用してコンテナーでコマンドを実行することはできません。Webhook を削除すると、**oc exec** コマンドを再び使用できるようになります。

4. 以下のコマンドを実行して依然として存在しているサービス、シークレット、および設定マップを削除します。

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

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

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

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

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

5. 以下のコマンドを実行して変更または検証用 webhook 設定を削除します。

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

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

6.3.2. Web コンソールでの Operator のアンインストール

手順

1. Web コンソールの **Administrator** パースペクティブで、**Operators → Installed Operators** に移動します。
2. フィルター一覧をスクロールするか、または **Filter by name** ボックスにキーワードを入力して **Web 端末** Operator を見つけます。
3. Web 端末 Operator の Options メニュー をクリックし、**Uninstall Operator** を選択します。
4. **Uninstall Operator** 確認ダイアログボックスで、**Uninstall** をクリックし、Operator、Operator デプロイメント、および Pod をクラスターから削除します。この Operator は実行を停止し、更新を受信しなくなります。

第7章 OPENSHIFT CONTAINER PLATFORM の WEB コンソールの無効化

OpenShift Container Platform の Web コンソールを無効にすることができます。

7.1. 前提条件

- OpenShift Container Platform クラスターをデプロイします。

7.2. WEB コンソールの無効化

console.config.openshift.io リソースを編集して Web コンソールを無効にすることができます。

- **console.operator.openshift.io** リソースを編集します。

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

以下の例は、変更できるリソースのパラメーターを表示しています。

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

(1) **managementState** パラメーター値を **Removed** に設定し、Web コンソールを無効にします。このパラメーターの他の有効な値には以下が含まれます。**Managed** ではクラスターの制御下でコンソールを有効にし、**Unmanaged** は Web コンソール管理を制御するのがユーザーであることを意味します。