Red Hat OpenShift Service on AWS 4

웹 콘솔

AWS의 Red Hat OpenShift Service에서 웹 콘솔 시작하기

Last Updated: 2024-11-03

Red Hat OpenShift Service on AWS 4 웹 콘솔

AWS의 Red Hat OpenShift Service에서 웹 콘솔 시작하기

법적 공지

초록

이 문서에서는 AWS 웹 콘솔에서 OpenShift 서비스에 액세스하고 사용자 정의하는 방법을 설명합니다.

차례

1장. 웹 콘솔 개요

AWS 웹 콘솔의 Red Hat OpenShift Service는 그래픽 사용자 인터페이스를 제공하여 프로젝트 데이터를 시각화하고 관리, 문제 해결 작업을 수행합니다. 웹 콘솔은 openshift-console 프로젝트의 컨트롤 플레인 노드에서 Pod로 실행됩니다. **console-operator** Pod에서 관리합니다. **관리자** 및 **개발자** 화면이 모두 지원됩니다.

관리자 및 **개발자 화면을** 통해 AWS에서 Red Hat OpenShift Service에 대한 퀵 스타트 튜토리얼을 생성할 수 있습니다. 퀵 스타트는 사용자 작업에 대한 가이드 튜토리얼이며 애플리케이션, Operator 또는 기타 제품 오퍼링을 사용하는 데 유용합니다.

1.1. 웹 콘솔의 관리자 화면 정보

관리자 화면을 사용하면 클러스터 인벤토리, 용량, 일반 및 특정 사용률 정보, 중요한 이벤트 스트림을 볼 수 있으므로 모두 계획 및 문제 해결 작업을 단순화할 수 있습니다. 프로젝트 관리자와 클러스터 관리자는 모두 **관리자** 화면을 볼 수 있습니다.

클러스터 관리자는 AWS 4.7 이상에서 Red Hat OpenShift Service에서 웹 터미널 Operator를 사용하여 포함된 명령행 터미널 인스턴스를 열 수도 있습니다.

참고

표시되는 기본 웹 콘솔 화면 정보는 사용자의 역할에 따라 다르게 표시됩니다. 사용자가 관리자로 인식되면 기본적으로 **관리자** 화면이 표시됩니다.

관리자 화면은 다음과 같이 관리자의 유스 케이스에 특정한 워크 플로우를 제공합니다.

- 워크로드, 스토리지, 네트워킹 및 클러스터 설정을 관리합니다.
- Operator Hub를 사용하여 Operator를 설치하고 관리합니다.
- 역할 및 역할 바인딩을 통해 사용자가 로그인하고 사용자 액세스를 관리할 수 있는 ID 공급자를 추가합니다.
- 클러스터 업데이트, 부분 클러스터 업데이트, 클러스터 Operator, CRD(사용자 정의 리소스 정의), 역할 바인딩, 리소스 할당량과 같은 다양한 고급 설정을 보고 관리합니다.
- 메트릭, 경고, 모니터링 대시보드와 같은 모니터링 기능에 액세스하고 관리합니다.
- 클러스터에 대한 로깅, 지표 및 높은 상태 정보를 보고 관리합니다.
- AWS의 Red Hat OpenShift Service에서 **관리자** 관점과 관련된 애플리케이션, 구성 요소 및 서비스와 시각적으로 상호 작용합니다.

1.2. 웹 콘솔의 개발자 화면 정보

개발자 화면은 애플리케이션, 서비스 및 데이터베이스를 배포하는 몇 가지 기본 제공 방법을 제공합니다. **개발자** 화면에서 다음을 수행할 수 있습니다.

- 구성요소의 롤링 및 재생성 롤아웃을 실시간으로 시각화합니다.
- 애플리케이션 상태, 리소스 사용률, 프로젝트 이벤트 스트리밍 및 할당량 사용을 확인합니다.
- 귀하의 프로젝트를 다른 사람들과 공유하십시오.

- 프로젝트에 대한 Prometheus Query Language(PromQL) 쿼리를 실행하고 플롯에 시각화된 메트릭을 검사하여 애플리케이션의 문제를 해결합니다. 메트릭은 클러스터 상태 및 모니터링 중인 사용자 정의 워크로드에 대한 정보를 제공합니다.

클러스터 관리자는 AWS 4.7 이상에서 Red Hat OpenShift Service의 웹 콘솔에서 포함된 명령행 터미널 인스턴스를 열 수도 있습니다.

참고

표시되는 기본 웹 콘솔 화면 정보는 사용자의 역할에 따라 다르게 표시됩니다. 사용자가 개발자로 인식되면 기본적으로 **개발자** 화면 정보가 표시됩니다.

개발자 화면은 다음과 같은 개발자의 유스 케이스에 특정한 워크 플로우를 제공합니다.

- 기존 코드베이스, 이미지 및 컨테이너 파일을 가져와 AWS에서 Red Hat OpenShift Service에서 애플리케이션을 생성하고 배포합니다.
- 프로젝트에서 관련 애플리케이션, 구성 요소 및 서비스와 시각적으로 상호 작용하고 배포 및 빌드 상태를 모니터링합니다.
- 애플리케이션에서 구성 요소를 그룹화하고 애플리케이션 내부 및 애플리케이션간에 구성 요소를 연결합니다.
- Serverless 기능 (기술 프리뷰)을 통합합니다.
- Eclipse Che를 사용하여 애플리케이션 코드를 편집할 수 있는 작업 공간을 만듭니다.

토폴로지 보기를 사용하여 프로젝트의 애플리케이션, 구성 요소 및 워크로드를 표시할 수 있습니다. 프로젝트에 워크로드가 없는 경우 **토폴로지** 보기에 생성하거나 가져올 일부 링크가 표시됩니다. **빠른 검색을** 사용하여 구성 요소를 직접 가져올 수도 있습니다.

추가 리소스

개발자 화면에서 토폴로지 보기를 사용하는 방법에 대한 자세한 내용은 **토폴로지** 보기를 사용하여 애플리케이션 구성 보기를 참조하십시오.

1.3. 화면 액세스

다음과 같이 웹 콘솔에서 **관리자** 및 **개발자** 화면에 액세스할 수 있습니다.

사전 요구 사항

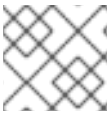
화면에 액세스하려면 웹 콘솔에 로그인해야 합니다. 기본 화면은 사용자 권한에 따라 자동으로 결정됩니다. 모든 프로젝트에 액세스할 수 있는 사용자에 대해 **관리자** 화면이 선택되지만, 자체 프로젝트에 대한 액세스 권한이 제한된 사용자는 **개발자** 화면이 선택됩니다.

추가 리소스

관점 변경에 대한 자세한 내용은 사용자 기본 설정 추가 를 참조하십시오.

프로세스

1. 화면 전환 기능을 사용하여 **관리자** 또는 **개발자** 화면으로 전환합니다.
2. **프로젝트** 드롭다운 목록에서 기존 프로젝트를 선택합니다. 이 드롭다운에서 새 프로젝트를 생성할 수도 있습니다.



참고

화면 전환기를 **cluster-admin** 으로만 사용할 수 있습니다.

추가 리소스

- 클러스터 정보 보기
- 웹 터미널 사용
- 퀵 스타트 튜토리얼 만들기

2장. 웹 콘솔에 액세스

AWS 웹 콘솔의 Red Hat OpenShift Service는 웹 브라우저에서 액세스할 수 있는 사용자 인터페이스입니다. 개발자는 웹 콘솔을 사용하여 프로젝트의 내용을 시각적으로 탐색 및 관리할 수 있습니다.

2.1. 전제 조건

- 웹 콘솔을 사용하려면 JavaScript가 활성화되어 있어야합니다. WebSockets을 지원하는 웹 브라우저를 사용하는 것이 좋습니다.
- 클러스터에 대한 지원 인프라를 작성하기 전에 OpenShift Container Platform 4.x Tested Integrations 페이지를 확인합니다.

2.2. 웹 콘솔 이해 및 액세스

웹 콘솔은 컨트롤 플레인 노드에서 Pod로 실행됩니다. Pod에서는 웹 콘솔을 실행하는 데 필요한 정적 환경을 제공합니다.

프로세스

1. OpenShift Cluster Manager 에 로그인하고 클러스터 이름을 클릭합니다.
2. 클러스터 **개요** 탭에서 **콘솔 열기** 를 클릭하고 인증 정보를 사용하여 로그인합니다.

또는 **oc whoami --show-console** 명령을 사용하여 웹 콘솔 URL을 가져옵니다.

3장. AWS 대시보드에서 RED HAT OPENSHIFT SERVICE를 사용하여 클러스터 정보 가져오기

AWS 웹 콘솔의 Red Hat OpenShift Service는 클러스터에 대한 높은 수준의 정보를 캡처합니다.

3.1. AWS 대시보드 페이지의 RED HAT OPENSHIFT SERVICE 정보

AWS 웹 콘솔의 Red Hat OpenShift Service에서 **홈 → 개요** 로 이동하여 클러스터에 대한 고급 정보를 캡처하는 AWS 대시보드에서 Red Hat OpenShift Service에 액세스합니다.

AWS 대시보드의 Red Hat OpenShift Service는 개별 대시보드 카드에 캡처되는 다양한 클러스터 정보를 제공합니다.

AWS 대시보드의 Red Hat OpenShift Service는 다음 카드로 구성됩니다.

- **Details**는 클러스터 정보에 대한 간략한 개요를 표시합니다.
  상태에는 **ok**, **error**, **warning**, **progress** 및 **unknown**이 포함되어 있습니다. 리소스는 사용자 정의 상태 이름을 추가 할 수 있습니다.
  - 클러스터 ID
  - 공급자
  - 버전
- **Cluster Inventory**는 리소스 수 및 관련 상태를 정보를 표시합니다. 이러한 정보는 문제 해결에 개입이 필요한 경우 매우 유용하며 다음과 같은 관련 정보가 포함되어 있습니다.
  - 노드 수
  - Pod 수
  - 영구 스토리지 볼륨 요청
  - 상태에 따라 나열된 클러스터의 베어 메탈 호스트 (**metal3** 환경에서만 사용 가능)
- **Status** 는 관리자가 클러스터 리소스를 사용하는 방법을 이해하는 데 도움이 됩니다. 리소스를 클릭하면 지정된 클러스터 리소스(CPU, 메모리 또는 스토리지)를 가장 많이 사용하는 Pod와 노드가 나열된 세부 정보 페이지로 이동합니다.
- **클러스터 사용률** 은 관리자가 다음과 같은 정보를 포함하여 높은 리소스 소비의 규모와 빈도를 이해하는 데 도움이 되도록 지정된 기간 동안 다양한 리소스의 용량을 표시합니다.
  - CPU 시간
  - 메모리 할당
  - 소비된 스토리지
  - 소비된 네트워크 리소스
  - Pod 수
- 활동에는 Pod 생성 또는 다른 호스트로의 가상 머신 마이그레이션과 같은 클러스터의 최근 활동과 관련된 메시지가 나열됩니다.

3.2. 리소스 및 프로젝트 제한 및 할당량 인식

웹 콘솔 개발자 화면의 토폴로지 보기에서 사용 가능한 리소스의 그래픽 표시를 볼 수 있습니다.

리소스에 리소스 제한이나 할당량에 대한 메시지가 있는 경우 리소스 이름 주위에 노란색 테두리가 표시됩니다. 리소스를 클릭하여 측면 패널을 열어 메시지를 확인합니다. 토폴로지 보기가 축소된 경우 노란색 점이 메시지를 사용할 수 있음을 나타냅니다.

보기 단축 메뉴에서 목록 보기를 사용하는 경우 리소스가 목록으로 표시됩니다.알림 열은 메시지를 사용할 수 있는지 여부를 나타냅니다.

4장. 동적 플러그인

4.1. 동적 플러그인 개요

4.1.1. 동적 플러그인 정보

동적 플러그인은 런타임 시 원격 소스에서 로드되고 해석됩니다. 동적 플러그인을 콘솔에 제공하고 노출하는 한 가지 방법은 OLM Operator를 사용하는 것입니다. Operator는 HTTP 서버와 함께 플랫폼에 배포를 생성하여 플러그인을 호스팅하고 Kubernetes 서비스를 사용하여 노출합니다.

동적 플러그인을 사용하면 런타임 시 콘솔 사용자 인터페이스에 사용자 정의 페이지 및 기타 확장을 추가할 수 있습니다. **ConsolePlugin** 사용자 정의 리소스는 콘솔에 플러그인을 등록하고 클러스터 관리자는 콘솔 Operator 구성에서 플러그인을 활성화합니다.

4.1.2. 주요 기능

동적 플러그인을 사용하면 AWS 환경에서 Red Hat OpenShift Service에 다음과 같은 사용자 지정을 수행할 수 있습니다.

- 사용자 지정 페이지를 추가합니다.
- 관리자 및 개발자 이외의 화면을 추가합니다.
- 검색 항목을 추가합니다.
- 탭과 작업을 리소스 페이지에 추가합니다.

4.1.3. 일반 지침

플러그인을 생성할 때 다음 일반 지침을 따르십시오.

- Node.js 및 yarn 은 플러그인을 빌드하고 실행하는 데 필요합니다.
- 충돌을 방지하려면 CSS 클래스 이름을 플러그인 이름으로 접두사로 지정합니다. 예를 들면 **my-plugin__heading** 및 **my-plugin__icon** 입니다.
- 다른 콘솔 페이지에서 일관된 모양, 느낌 및 동작을 유지합니다.
- 플러그인을 생성할 때 react-i18next 현지화 지침을 따르십시오. 다음 예제에서와 같이 **useTranslation** 후크를 사용할 수 있습니다.

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

- 요소 선택기와 같은 플러그인 구성 요소 외부의 DestinationRule에 영향을 줄 수 있는 선택기를 사용하지 마십시오. 이는 API가 아니며 변경될 수 있습니다. 이를 사용하면 플러그인이 손상될 수 있습니다. 플러그인 구성 요소 외부에서 4.6.1에 영향을 줄 수 있는 요소 선택기와 같은 선택기를 사용하지 마십시오.
- 플러그인 웹 서버에서 제공하는 모든 자산에 대해 **Content-Type** 응답 헤더를 사용하여 유효한 JavaScript Multipurpose Internet Mail Extension(MIME) 유형을 제공합니다. 각 플러그인 배포에는 지정된 플러그인의 생성된 자산을 호스팅하는 웹 서버가 포함되어야 합니다.

- Webpack 버전 5 이상을 사용하여 Webpack 플러그인을 빌드해야 합니다.
- 충돌을 방지하려면 CSS 클래스 이름 앞에 플러그인 이름을 붙여야 합니다. 예를 들면 **my-plugin__heading** 및 **my-plugin___icon** 입니다.
- 다른 콘솔 페이지와 일관된 모양, 느낌 및 동작을 유지해야 합니다.
- 요소 선택기와 같이 플러그인 구성 요소 외부의 태그에 영향을 줄 수 있는 선택기를 피해야 합니다. 이는 API가 아니며 변경될 수 있습니다.
- 플러그인 웹 서버에서 제공하는 모든 자산에 대해 **Content-Type** 응답 헤더를 사용하여 유효한 JavaScript Multipurpose Internet Mail Extension(MIME) 유형을 제공해야 합니다. 각 플러그인 배포에는 지정된 플러그인의 생성된 자산을 호스팅하는 웹 서버가 포함되어야 합니다.

PatternFly 지침

플러그인을 생성할 때 PatternFly 사용에 대한 다음 지침을 따르십시오.

- PatternFly 구성 요소 및 PatternFly CSS 변수를 사용합니다. 핵심 PatternFly 구성 요소는 SDK를 통해 사용할 수 있습니다. PatternFly 구성 요소 및 변수를 사용하면 플러그인이 향후 콘솔 버전에서 일관성을 유지하는 데 도움이 됩니다.
  - AWS 버전 4.14 및 이전 버전에서 Red Hat OpenShift Service를 사용하는 경우 Patternfly 4.x를 사용하십시오.
  - AWS 4.15 이상에서 Red Hat OpenShift Service를 사용하는 경우 Patternfly 5.x를 사용하십시오.
- PatternFly의 접근성 기본 사항에 따라 플러그인에 액세스할 수 있도록 합니다.
- Bootstrap 또는 Tailwind와 같은 다른 CSS 라이브러리는 사용하지 마십시오. PatternFly와 충돌할 수 있으며 콘솔의 나머지 부분과 일치하지 않을 수 있습니다. 플러그인은 기본 PatternFly 스타일에서 평가할 사용자 인터페이스에 고유한 스타일만 포함해야 합니다. 플러그인의 **@patternfly/react-styles/*/.css** 또는 모든 스타일에서 **@patternfly/patternfly** 패키지와 같은 스타일을 가져오지 마십시오.
- 콘솔 애플리케이션은 지원되는 모든 PatternFly 버전에 대한 기본 스타일을 로드합니다.

4.1.3.1. react-i18 next로 메시지 번역

플러그인 템플릿 은 react-i18 next 를 사용하여 메시지를 변환하는 방법을 보여줍니다.

사전 요구 사항

- 플러그인 템플릿이 로컬에 복제되어 있어야 합니다.
- 선택 사항: 플러그인을 로컬에서 테스트하려면 컨테이너에서 AWS 웹 콘솔에서 Red Hat OpenShift Service를 실행합니다. Docker 또는 Podman 3.2.0 이상을 사용할 수 있습니다.

프로세스

1. 이름 충돌을 방지하려면 name에 **plugin__** 접두사를 추가합니다. 플러그인 템플릿은 기본적으로 **plugin__console-plugin-template** 네임스페이스를 사용하며, **plugin__my-plugin** 과 같은 플러그인 이름을 바꿀 때 업데이트해야 합니다. **useTranslation** 후크를 사용할 수 있습니다. 예를 들면 다음과 같습니다.

```
conster Header: React.FC = () => {
```

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

중요

i18n 네임스페이스와 ConsolePlugin 리소스 이름을 일치시켜야 합니다.

2. 필요한 동작을 기반으로 spec.i18n.loadType 필드를 설정합니다.

예 4.1. plugin__console-demo-plugin

```
spec:
  backend:
    service:
      basePath: /
      name: console-demo-plugin
      namespace: console-demo-plugin
      port: 9001
    type: Service
  displayName: OpenShift Console Demo Plugin
  i18n:
    loadType: Preload (1)
```

(1) 로드 중 동적 플러그인 후 i18n 네임스페이스에서 모든 플러그인의 현지화 리소스를 로드합니다.

3. console-extensions.json 의 라벨에 %plugin__console-plugin-template~My Label% 형식을 사용합니다. 콘솔은 plugin__console-plugin-template 네임스페이스에서 현재 언어의 메시지로 값을 대체합니다. 예를 들면 다음과 같습니다.

```
{
  "type": "console.navigation/section",
  "properties": {
    "id": "admin-demo-section",
    "perspective": "admin",
    "name": "%plugin__console-plugin-template~Plugin Template%"
  }
}
```

4. i18next-parser 의 TypeScript 파일에 주석을 포함하여console-extensions.json 의 메시지를 메시지 카탈로그에 추가합니다. 예를 들면 다음과 같습니다.

```
// t('plugin__console-demo-plugin~Demo Plugin')
```

5. 메시지를 추가하거나 변경할 때 플러그인 템플릿의 locales 폴더에 있는 JSON 파일을 업데이트하려면 다음 명령을 실행합니다.

```
$ yarn i18n
```

4.2. 동적 플러그인 시작하기

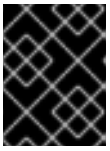
동적 플러그인을 사용하여 시작하려면 AWS 동적 플러그인에 새 Red Hat OpenShift Service를 작성하도록 환경을 설정해야 합니다. 새 플러그인을 작성하는 방법의 예는 Pod 페이지에 탭 추가를 참조하십시오.

4.2.1. 동적 플러그인 개발

로컬 개발 환경을 사용하여 플러그인을 실행할 수 있습니다. AWS 웹 콘솔의 Red Hat OpenShift Service는 로그인한 클러스터에 연결된 컨테이너에서 실행됩니다.

사전 요구 사항

- 플러그인 생성을 위한 템플릿이 포함된 **console-plugin-template** 리포지토리를 복제해야 합니다.



중요

Red Hat은 사용자 정의 플러그인 코드를 지원하지 않습니다. 해당 플러그인에 대해 협업 커뮤니티 지원 만 사용할 수 있습니다.

- AWS 클러스터에 Red Hat OpenShift Service가 실행되고 있어야 합니다.
- OpenShift CLI(**oc**)가 설치되어 있어야 합니다.
- **yarn** 이 설치되어 있어야 합니다.
- Docker v3.2.0 이상 또는 Podman v3.2.0 이상이 설치되어 실행되고 있어야 합니다.

프로세스

1. 두 개의 터미널 창을 엽니다.
2. 하나의 터미널 창에서 다음 명령을 실행하여 yarn을 사용하여 플러그인의 종속 항목을 설치합니다.

```
$ yarn install
```

3. 설치 후 다음 명령을 실행하여 yarn을 시작합니다.

```
$ yarn run start
```

4. 다른 터미널 창에서 CLI를 통해 AWS의 Red Hat OpenShift Service에 로그인합니다.

```
$ oc login
```

5. 다음 명령을 실행하여 로그인한 클러스터에 연결된 컨테이너의 AWS 웹 콘솔에서 Red Hat OpenShift Service를 실행합니다.

```
$ yarn run start-console
```

참고

yarn run start-console 명령은 **amd64** 이미지를 실행하고 Apple Silicon 및 Podman으로 실행하면 실패할 수 있습니다. 다음 명령을 실행하여 **qemu-user-static** 로 이 문제를 해결할 수 있습니다.

```
$ podman machine ssh
$ sudo -i
$ rpm-ostree install qemu-user-static
$ systemctl reboot
```

검증

- 실행 중인 플러그인을 확인하려면 localhost:9000 으로 이동하십시오. **window.SERVER_FLAGS.consolePlugins** 값을 검사하여 런타임 시 로드되는 플러그인 목록을 확인합니다.

4.3. 클러스터에 플러그인 배포

AWS 클러스터의 Red Hat OpenShift Service에 플러그인을 배포할 수 있습니다.

4.3.1. Docker로 이미지 빌드

클러스터에 플러그인을 배포하려면 이미지를 빌드하고 먼저 이미지 레지스트리로 푸시해야 합니다.

프로세스

1. 다음 명령을 사용하여 이미지를 빌드합니다.

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

2. 선택 사항: 이미지를 테스트하려면 다음 명령을 실행합니다.

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

3. 다음 명령을 실행하여 이미지를 내보냅니다.

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

4.3.2. 클러스터에 플러그인 배포

레지스트리에 변경 사항이 있는 이미지를 푸시한 후 Helm 차트를 사용하여 클러스터에 플러그인을 배포할 수 있습니다.

사전 요구 사항

- 이전에 내보낸 플러그인이 포함된 이미지의 위치가 있어야 합니다.

참고

플러그인의 필요에 따라 추가 매개변수를 지정할 수 있습니다. values.yaml 파일은 지원되는 전체 매개변수 세트를 제공합니다.

프로세스

1. 플러그인 이름을 사용하여 클러스터에 플러그인을 배포하려면 Helm 릴리스 이름으로 **-n** 명령줄 옵션에 지정된 기존 네임스페이스 또는 기존 네임스페이스에 Helm 차트를 설치합니다. 다음 명령을 사용하여 **plugin.image** 매개변수 내에 이미지 위치를 제공합니다.

```
$ helm upgrade -i  my-plugin charts/openshift-console-plugin -n my-plugin-
namespace --create-namespace --set plugin.image=my-plugin-image-location
```

다음과 같습니다.

n <my-plugin-namespace>

플러그인을 배포할 기존 네임스페이스를 지정합니다.

--create-namespace

선택 사항: 새 네임스페이스에 배포하는 경우 이 매개변수를 사용합니다.

--set plugin.image=my-plugin-image-location

plugin.image 매개변수 내의 이미지 위치를 지정합니다.

참고

AWS 4.10 이상에서 Red Hat OpenShift Service에 배포하는 경우 **--set plugin.securityContext.enabled=false** 를 추가하여 Pod 보안과 관련된 구성을 제외하는 것이 좋습니다.

2. 선택 사항: **charts/openshift-console-plugin/values.yaml** 파일에서 지원되는 매개변수 세트를 사용하여 추가 매개변수를 지정할 수 있습니다.

```
plugin:
  name: ""
  description: ""
  image: ""
  imagePullPolicy: IfNotPresent
  replicas: 2
  port: 9443
  securityContext:
    enabled: true
  podSecurityContext:
    enabled: true
    runAsNonRoot: true
    seccompProfile:
      type: RuntimeDefault
  containerSecurityContext:
    enabled: true
    allowPrivilegeEscalation: false
    capabilities:
      drop:
        - ALL
  resources:
    requests:
      cpu: 10m
      memory: 50Mi
  basePath: /
  certificateSecretName: ""
```

```
  serviceAccount:
    create: true
    annotations: {}
    name: ""
  patcherServiceAccount:
    create: true
    annotations: {}
    name: ""
  jobs:
    patchConsoles:
      enabled: true
      image: "registry.redhat.io/openshift4/ose-tools-
rhel8@sha256:e44074f21e0cca6464e50cb6ff934747e0bd11162ea01d522433a1a1ae1161
03"
      podSecurityContext:
        enabled: true
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
      containerSecurityContext:
        enabled: true
        allowPrivilegeEscalation: false
        capabilities:
          drop:
            - ALL
      resources:
        requests:
          cpu: 10m
          memory: 50Mi
```

검증

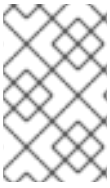
- 관리 → 클러스터 설정 → 구성 → 콘솔 operator.openshift.io → 콘솔 플러그인에서 이동하거나 개요 페이지를 방문하여 활성화된 플러그인 목록을 확인합니다.

참고

새 플러그인 구성이 표시되는 데 몇 분 정도 걸릴 수 있습니다. 플러그인이 표시되지 않는 경우 플러그인이 최근에 활성화된 경우 브라우저를 새로 고쳐야 할 수 있습니다. 런타임 시 오류가 발생하면 브라우저 개발자 도구의 JS 콘솔을 확인하여 플러그인 코드의 오류를 확인합니다.

4.3.3. 플러그인 서비스 프록시

플러그인에서 클러스터 내 서비스에 HTTP 요청을 수행해야 하는 경우 **spec.proxy** array 필드를 사용하여 **ConsolePlugin** 리소스에서 서비스 프록시를 선언할 수 있습니다. 콘솔 백엔드는 **/api/proxy/plugin/<plugin-name>/<proxy-alias>/<request-path>?<optional-query-parameters** > 끝점을 노출하여 플러그인과 서비스 간의 통신을 프록시합니다. 프록시 요청은 기본적으로 *서비스 CA 번들을* 사용합니다. 서비스에서 HTTPS를 사용해야 합니다.



참고

플러그인은 JavaScript 코드에서 요청을 만들기 위해 **consolefetch** API를 사용해야 합니다. 그렇지 않으면 일부 요청이 실패할 수 있습니다. 자세한 내용은 "Dynamic plugin API"를 참조하십시오.

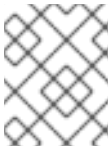
각 항목에 대해 끝점 및 별칭 필드 아래에 프록시의 끝점 과 별칭 을 지정해야 합니다. Service 프록시 유형의 경우 끝점 유형 필드를 **Service** 로 설정해야 하며서비스에 이름,네임스페이스, 포트 필드의 값이 포함되어야 합니다. 예를 들어 **/api/proxy/plugin/helm/ helm-charts /release=10** 은 10개의 helm 릴리스를 나열하는 **helm** -charts 서비스가 있는 helm 플러그인의 프록시 요청 경로입니다.

서비스 프록시 예

```
apiVersion: console.openshift.io/v1
kind: ConsolePlugin
metadata:
  name:<plugin-name>
spec:
  proxy:
  - alias: helm-charts 1
    authorization: UserToken 2
    caCertificate: '-----BEGIN CERTIFICATE-----\nMIID....'en 3
    endpoint: 4
      service:
        name: <service-name>
        namespace: <service-namespace>
        port: <service-port>
      type: Service
```

1 프록시의 별칭입니다.

2 서비스 프록시 요청에 AWS 액세스 토큰에 로그인한 사용자의 Red Hat OpenShift Service가 포함되어야 하는 경우 권한 부여 필드를 **UserToken** 으로 설정해야 합니다.



참고

서비스 프록시 요청에 로그인한 사용자의 Red Hat OpenShift Service on AWS 액세스 토큰이 포함되어 있지 않은 경우 권한 부여 필드를 **None** 으로 설정합니다.

3 서비스에서 사용자 정의 서비스 CA를 사용하는 경우 **caCertificate** 필드에 인증서 번들이 포함되어야 합니다.

4 프록시의 끝점입니다.

4.3.4. 브라우저에서 플러그인 비활성화

콘솔 사용자는 **disable-plugins** 쿼리 매개변수를 사용하여 일반적으로 런타임 시 로드되는 특정 또는 모든 동적 플러그인을 비활성화할 수 있습니다.

프로세스

- 특정 플러그인을 비활성화하려면 플러그인 이름의 쉼표로 구분된 목록에서 비활성화하려는 플러그인을 제거합니다.
- 모든 플러그인을 비활성화하려면 **disable-plugins** 쿼리 매개변수에 빈 문자열을 남겨 둡니다.

참고

클러스터 관리자는 웹 콘솔의 클러스터 설정 페이지에서 플러그인을 비활성화할 수 있습니다.

4.4. 동적 플러그인 예

예제를 수행하기 전에 동적 플러그인 개발단계에 따라 플러그인이 작동하는지 확인합니다.

4.4.1. Pod 페이지에 탭 추가

AWS 웹 콘솔에서 Red Hat OpenShift Service에 사용자 지정할 수 있는 다양한 사용자 지정이 있습니다. 다음 절차에서는 플러그인의 확장 예제로 Pod 세부 정보 페이지에 탭을 추가합니다.

참고

AWS 웹 콘솔의 Red Hat OpenShift Service는 로그인한 클러스터에 연결된 컨테이너에서 실행됩니다. 자체 만들기 전에 플러그인을 테스트하는 정보는 "동적 플러그인 개발"을 참조하십시오.

프로세스

1. 새 탭에서 플러그인을 생성하는 템플릿이 포함된 **console-plugin-template** 리포지터리를 방문하십시오.

중요

Red Hat에서는 사용자 정의 플러그인 코드를 지원하지 않습니다. 해당 플러그인에 대해 협업 커뮤니티 지원 만 사용할 수 있습니다.

2. 이 템플릿 사용 → *새 리포지토리 만들기*를 클릭하여 템플릿의 GitHub 리포지토리를 생성합니다.
3. 새 리포지토리의 이름을 플러그인 이름으로 변경합니다.
4. 코드를 편집할 수 있도록 새 리포지토리를 로컬 시스템에 복제합니다.
5. **consolePlugin** 선언에 플러그인 메타데이터를 추가하여**package.json** 파일을 편집합니다. 예를 들면 다음과 같습니다.

```
"consolePlugin": {
  "name": "my-plugin", (1)
  "version": "0.0.1", (2)
  "displayName": "My Plugin", (3)
  "description": "Enjoy this shiny, new console plugin!", (4)
  "exposedModules": {
    "ExamplePage": "./components/ExamplePage"
  },
  "dependencies": {
    "@console/pluginAPI": "/*"
  }
}
```

(1) 플러그인 이름을 업데이트합니다.

2. 버전을 업데이트합니다.
3. 플러그인의 표시 이름을 업데이트합니다.
4. 플러그인에 대한 개요로 설명을 업데이트합니다.

6. **console-extensions.json** 파일에 다음을 추가합니다.

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

7. 다음 변경 사항을 포함하도록 **package.json** 파일을 편집합니다.

```
    "exposedModules": {
      "ExamplePage": "./components/ExamplePage",
      "ExampleTab": "./components/ExampleTab"
    }
```

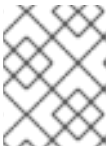
8. 새 파일ECDHE **/components/ECDHETab.tsx**를 생성하고 다음 스크립트를 추가하여Pod 페이지의 새 사용자 지정 탭에 표시되는 메시지를 작성합니다.

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

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

9. 플러그인 이름과 함께 Helm 차트를 새 네임스페이스 또는 **-n** 명령줄 옵션에 지정된 기존 네임스페이스에 Helm 릴리스 이름으로 설치하여 클러스터에 플러그인을 배포합니다. 다음 명령을 사용하여 **plugin.image** 매개변수 내에 이미지 위치를 제공합니다.

```
$ helm upgrade -i  my-plugin charts/openshift-console-plugin -n my-plugin-
namespace --create-namespace --set plugin.image=my-plugin-image-location
```



참고

클러스터에 플러그인을 배포하는 방법에 대한 자세한 내용은 "클러스터에 플러그인 배포"를 참조하십시오.

검증

- Pod 페이지로 이동하여 추가된 탭을 확인합니다.

4.5. 동적 플러그인 참조

플러그인을 사용자 지정할 수 있는 확장 기능을 추가할 수 있습니다. 그런 다음 이러한 확장은 런타임 시 콘솔에 로드됩니다.

4.5.1. 동적 플러그인 확장 유형

console.action/filter
ActionFilter 를 사용하여 작업을 필터링할 수 있습니다.

이름	값 유형	선택 사항	설명
contextId	**string**	제공되지 않음	컨텍스트 ID를 사용하면 애플리케이션의 특정 영역에 기여된 작업의 범위를 좁히는 데 도움이 됩니다. 예를 들면 **topology** 및 **helm**이 있습니다.
filter	**CodeRef<(scope: any, action: Action) ECDHE boolean>**	제공되지 않음	일부 조건에 따라 작업을 필터링하는 함수입니다. **scope**: 작업을 제공해야 하는 범위입니다. HPA(수평 Pod 자동 스케일러)를 사용하여 배포에서 **ModifyCount** 작업을 제거하려면 후크가 필요할 수 있습니다.

console.action/group
ActionGroup 은 하위 메뉴일 수도 있는 작업 그룹을 생성합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	작업 섹션을 식별하는 데 사용되는 ID입니다.
label	**string**	제공됨	UI에 표시할 레이블입니다. 하위 메뉴에 필요합니다.
submenu	**boolean**	제공됨	이 그룹을 하위 메뉴로 표시할지 여부입니다.

이름	값 유형	선택 사항	설명
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 값이 우선합니다.

console.action/provider
ActionProvider는 특정 컨텍스트에 대한 작업 목록을 반환하는 후크를 생성합니다.

이름	값 유형	선택 사항	설명
contextId	**string**	제공되지 않음	컨텍스트 ID를 사용하면 애플리케이션의 특정 영역에 기여된 작업의 범위를 좁히는 데 도움이 됩니다. 예를 들면 **topology** 및 **helm**이 있습니다.
provider	**CodeRef<Extension Hook<Action[], any>>**	제공되지 않음	지정된 범위에 대한 작업을 반환하는 React 후크입니다. **contextId** = **resource** 인 경우 범위는 항상 Kubernetes 리소스 오브젝트입니다.

console.action/resource-provider
ResourceActionProvider 는 특정 리소스 모델에 대한 작업 목록을 반환하는 후크를 생성합니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sKindVersionModel**	제공되지 않음	이 공급자가 작업을 제공하는 모델입니다.
provider	**CodeRef<Extension Hook<Action[], any>>**	제공되지 않음	지정된 리소스 모델에 대한 작업을 반환하는 반응 후크

console.alert-action
이 확장 기능을 사용하면 rule.name 값을 기반으로 콘솔에서 특정 Prometheus 경고가 확인될 때 특정 작업을 트리거할 수 있습니다.

이름	값 유형	선택 사항	설명
경고	**string**	제공되지 않음	**alert.rule.name** 속성에서 정의한 경고 이름
text	**string**	제공되지 않음	
작업	**CodeRef<(alert: any) ⇒ void>**	제공되지 않음	부작용을 수행하는 기능

console.catalog/item-filter
이 확장은 플러그인에 특정 카탈로그 항목을 필터링할 수 있는 처리기를 제공하는 데 사용할 수 있습니다. 예를 들어 플러그인은 특정 공급자의 helm 차트를 필터링하는 필터를 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
catalogId	**string | string[]**	제공되지 않음	이 공급자가 제공하는 카탈로그의 고유 식별자입니다.
type	**string**	제공되지 않음	카탈로그 항목 유형의 ID를 입력합니다.
filter	**CodeRef<(item: CatalogItem) ⇒ boolean>**	제공되지 않음	특정 유형의 항목을 필터링합니다. 값은 **CatalogItem[]** 를 사용하고 필터 기준에 따라 하위 집합을 반환하는 함수입니다.

console.catalog/item-metadata
이 확장 기능을 사용하여 특정 카탈로그 항목에 메타데이터를 추가하는 공급자를 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
catalogId	**string | string[]**	제공되지 않음	이 공급자가 제공하는 카탈로그의 고유 식별자입니다.
type	**string**	제공되지 않음	카탈로그 항목 유형의 ID를 입력합니다.
provider	**CodeRef<ExtensionHook<CatalogItemMetadataProviderFunction, CatalogExtensionHookOptions>>**	제공되지 않음	특정 유형의 카탈로그 항목에 메타데이터를 제공하는 데 사용할 함수를 반환하는 후크입니다.

console.catalog/item-provider
이 확장을 통해 플러그인은 카탈로그 항목 유형에 대한 공급자를 제공할 수 있습니다. 예를 들어 Helm 플러그인은 모든 Helm 차트를 가져오는 공급자를 추가할 수 있습니다. 이 확장 기능을 사용하여 다른 플러그인에서 특정 카탈로그 항목 유형에 항목을 추가할 수도 있습니다.

이름	값 유형	선택 사항	설명
catalogId	**string | string[]**	제공되지 않음	이 공급자가 제공하는 카탈로그의 고유 식별자입니다.
type	**string**	제공되지 않음	카탈로그 항목 유형의 ID를 입력합니다.
title	**string**	제공되지 않음	카탈로그 항목 공급자의 제목
provider	**CodeRef<ExtensionHook<CatalogItem<any>[], CatalogExtensionHookOptions>>**	제공되지 않음	항목을 가져와서 카탈로그에 대해 정규화합니다. 값은 반응 효과 후크입니다.
priority	**number**	제공됨	이 공급자의 우선 순위입니다. 기본값은 **0** 입니다. 우선순위가 높은 공급자는 다른 공급자가 제공하는 카탈로그 항목을 재정의할 수 있습니다.

console.catalog/item-type
이 확장을 통해 플러그인은 새로운 유형의 카탈로그 항목을 제공할 수 있습니다. 예를 들어 Helm 플러그인은 새 카탈로그 항목 유형을 개발자 카탈로그에 기여하려는 HelmCharts로 정의할 수 있습니다.

이름	값 유형	선택 사항	설명
type	**string**	제공되지 않음	카탈로그 항목의 유형입니다.
title	**string**	제공되지 않음	카탈로그 항목의 이름입니다.
catalogDescription	**string | CodeRef<React.ReactNode>**	제공됨	유형별 카탈로그에 대한 설명입니다.
typeDescription	**string**	제공됨	카탈로그 항목 유형에 대한 설명입니다.

이름	값 유형	선택 사항	설명
filters	**CatalogItemAttribute []**	제공됨	카탈로그 항목별 사용자 지정 필터.
groupings	**CatalogItemAttribute []**	제공됨	카탈로그 항목별 사용자 지정 그룹화입니다.

console.catalog/item-type-metadata
이 확장을 통해 플러그인은 사용자 정의 필터 또는 카탈로그 항목 유형에 대한 그룹화와 같은 추가 메타데이터를 제공할 수 있습니다. 예를 들어 플러그인은 차트 공급자를 기반으로 필터링할 수 있는 HelmChart에 대한 사용자 정의 필터를 연결할 수 있습니다.

이름	값 유형	선택 사항	설명
type	**string**	제공되지 않음	카탈로그 항목의 유형입니다.
filters	**CatalogItemAttribute []**	제공됨	카탈로그 항목별 사용자 지정 필터.
groupings	**CatalogItemAttribute []**	제공됨	카탈로그 항목별 사용자 지정 그룹화입니다.

console.cluster-overview/inventory-item
새 인벤토리 항목을 클러스터 개요 페이지에 추가합니다.

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<{}>>**	제공되지 않음	렌더링할 구성 요소입니다.

console.cluster-overview/multiline-utilization-item
새 클러스터 개요 다중 라인 사용 항목을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	사용률 항목의 제목입니다.
getUtilizationQueries	**CodeRef<GetMultilineQueries>**	제공되지 않음	Prometheus 사용률 쿼리.

이름	값 유형	선택 사항	설명
humanize	**CodeRef<Humanize>**	제공되지 않음	Prometheus 데이터를 사람이 읽을 수 있는 형식으로 변환합니다.
TopConsumerPopovers	**CodeRef<React.ComponentType<TopConsumerPopoverProps>[]>**	제공됨	일반 값 대신 상위 소비자 팝업을 표시합니다.

console.cluster-overview/utilization-item
새 클러스터 개요 사용률 항목을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	사용률 항목의 제목입니다.
getUtilizationQuery	**CodeRef<GetQuery>**	제공되지 않음	Prometheus 사용률 쿼리.
humanize	**CodeRef<Humanize>**	제공되지 않음	Prometheus 데이터를 사람이 읽을 수 있는 형식으로 변환합니다.
getTotalQuery	**CodeRef<GetQuery>**	제공됨	Prometheus 총 쿼리.
getRequestQuery	**CodeRef<GetQuery>**	제공됨	Prometheus 요청 쿼리.
getLimitQuery	**CodeRef<GetQuery>**	제공됨	Prometheus 제한 쿼리.
TopConsumerPopover	**CodeRef<React.ComponentType<TopConsumerPopoverProps>>**	제공됨	일반 값 대신 상위 소비자 팝업을 표시합니다.

console.context-provider
웹 콘솔 애플리케이션 루트에 새 React 컨텍스트 공급자를 추가합니다.

이름	값 유형	선택 사항	설명
provider	**CodeRef<Provider<T>>**	제공되지 않음	컨텍스트 공급자 구성 요소.
useValueHook	**CodeRef<() ⇒ T>**	제공되지 않음	Context 값의 후크입니다.

console.dashboards/card
새 대시보드 카드를 추가합니다.

이름	값 유형	선택 사항	설명
tab	**string**	제공되지 않음	카드를 추가할 대시보드 탭의 ID입니다.
position	**'LEFT' | 'RIGHT' | 'MAIN'**	제공되지 않음	대시보드에 있는 카드의 그리드 위치입니다.
component	**CodeRef<React.ComponentType<{}>>**	제공되지 않음	대시보드 카드 구성 요소.
span	**OverviewCardSpan**	제공됨	열에 있는 카드의 수직 범위입니다. 작은 화면에 대해 무시되고 기본값은 **12** 입니다.

console.dashboards/custom/overview/detail/item
개요 대시보드의 세부 정보 카드에 항목을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	세부 정보 카드 제목
component	**CodeRef<React.ComponentType<{}>>**	제공되지 않음	개요 세부 정보 항목 구성 요소에서 렌더링한 값
valueClassName	**string**	제공됨	className의 값
isLoading	**CodeRef<() ⇒ boolean>**	제공됨	구성 요소의 로드 상태를 반환하는 함수
error	**CodeRef<() ⇒ string>**	제공됨	구성 요소에서 표시할 오류를 반환하는 함수

console.dashboards/overview/activity/resource
활동 트리거가 Kubernetes 리소스 감시를 기반으로 하는 개요 대시보드의 활동 카드에 활동을 추가합니다.

이름	값 유형	선택 사항	설명
k8sResource	**CodeRef<FireECDHEResource & { isList: true; }>**	제공되지 않음	교체할 사용률 항목입니다.

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<K8sActivityProps<T>>>**	제공되지 않음	작업 구성 요소입니다.
isActivity	**CodeRef<(resource: T) ⇒ boolean>**	제공됨	지정된 리소스가 작업을 나타내는지 여부를 결정하는 함수입니다. 정의되지 않은 경우 모든 리소스는 활동을 나타냅니다.
getTimestamp	**CodeRef<(resource: T) ⇒ Date>**	제공됨	지정된 동작에 대한 타임스탬프로, 주문에 사용됩니다.

console.dashboards/overview/health/operator
상태 소스가 Kubernetes REST API인 개요 대시보드의 상태 카드에 상태 하위 시스템을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	팝업 메뉴에 있는 Operator 섹션의 제목입니다.
resources	**CodeRef<FirehoseResource[]>**	제공되지 않음	**healthHandler** 로 가져와 전달할 Kubernetes 리소스입니다.
getOperatorsWithStatuses	**CodeRef<GetOperatorsWithStatuses<T>>**	제공됨	Operator의 상태를 확인합니다.
operatorRowLoader	**CodeRef<React.ComponentType<OperatorRowProps<T>>>**	제공됨	팝업 행 구성 요소의 로더입니다.
viewAllLink	**string**	제공됨	모든 리소스 페이지에 대한 링크입니다. 제공되지 않으면 prop 리소스의 첫 번째 리소스 목록 페이지가 사용됩니다.

console.dashboards/overview/health/prometheus
상태 소스가 Prometheus인 개요 대시보드의 상태 카드에 상태 하위 시스템을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	하위 시스템의 표시 이름입니다.
queries	**string[]**	제공되지 않음	Prometheus 쿼리입니다.
healthHandler	**CodeRef<PrometheusHealthHandler>**	제공되지 않음	하위 시스템의 상태를 해결합니다.
additionalResource	**CodeRef<FirehoseResource>**	제공됨	**healthHandler** 로 가져와 전달할 추가 리소스입니다.
popupComponent	**CodeRef<React.ComponentType<PrometheusHealthPopupProps>>**	제공됨	팝업 메뉴 콘텐츠의 로더입니다. 정의된 경우 상태 항목이 링크로 표시되고 지정된 콘텐츠가 있는 팝업 메뉴가 열립니다.
popupTitle	**string**	제공됨	팝업의 제목입니다.
disallowedControlPlaneTopology	**string[]**	제공됨	하위 시스템을 숨겨야 하는 컨트롤 플레인 토폴로지입니다.

console.dashboards/overview/health/resource
상태 소스가 Kubernetes 리소스인 개요 대시보드의 상태 카드에 상태 하위 시스템을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	하위 시스템의 표시 이름입니다.
resources	**CodeRef<WatchK8sResources<T>>**	제공되지 않음	**healthHandler** 로 가져와 전달할 Kubernetes 리소스입니다.
healthHandler	**CodeRef<ResourceHealthHandler<T>>**	제공되지 않음	하위 시스템의 상태를 해결합니다.
popupComponent	**CodeRef<WatchK8sResults<T>>**	제공됨	팝업 메뉴 콘텐츠의 로더입니다. 정의된 경우 상태 항목이 링크로 표시되고 지정된 콘텐츠가 있는 팝업 메뉴가 열립니다.
popupTitle	**string**	제공됨	팝업의 제목입니다.

console.dashboards/overview/health/url
상태 소스가 Kubernetes REST API인 개요 대시보드의 상태 카드에 상태 하위 시스템을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	하위 시스템의 표시 이름입니다.
url	**string**	제공되지 않음	데이터를 가져올 URL입니다. 기본 Kubernetes URL이 접두사로 지정됩니다.
healthHandler	**CodeRef<URLHealthHandler<T, K8sResourceCommon | K8sResourceCommon[]>>**	제공되지 않음	하위 시스템의 상태를 해결합니다.
additionalResource	**CodeRef<FirehoseResource>**	제공됨	**healthHandler**로 가져와 전달할 추가 리소스입니다.
popupComponent	**CodeRef<React.ComponentType<{ healthResult?: T; healthResultError?: any; k8sResult?: FirehoseResult<R>; }>>**	제공됨	팝업 컨텐츠용 로더입니다. 정의된 경우 상태 항목은 지정된 콘텐츠로 팝업을 여는 링크로 표시됩니다.
popupTitle	**string**	제공됨	팝업의 제목입니다.

console.dashboards/overview/inventory/item
개요 인벤토리 카드에 리소스 타일을 추가합니다.

이름	값 유형	선택 사항	설명
model	**CodeRef<T>**	제공되지 않음	가져올 **resource**의 모델입니다. 모델의 **label** 또는 **abbr**를 가져오는 데 사용됩니다.
mapper	**CodeRef<StatusGroupMapper<T, R>>**	제공됨	다양한 상태를 그룹에 매핑하는 함수입니다.
additionalResources	**CodeRef<WatchK8sResources<R>>**	제공됨	**mapper** 함수로 가져와 전달할 추가 리소스입니다.

console.dashboards/overview/inventory/item/group
인벤토리 상태 그룹을 추가합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	상태 그룹의 ID입니다.
icon	**CodeRef<React.ReactElement<any, string | React.JSXElementConstructor<any>>>**	제공되지 않음	상태 그룹 아이콘을 나타내는 반응 구성 요소입니다.

console.dashboards/overview/inventory/item/replacement
개요 인벤토리 카드를 대체합니다.

이름	값 유형	선택 사항	설명
model	**CodeRef<T>**	제공되지 않음	가져올 **resource**의 모델입니다. 모델의 **label** 또는 **abbr**를 가져오는 데 사용됩니다.
mapper	**CodeRef<StatusGroupMapper<T, R>>**	제공됨	다양한 상태를 그룹에 매핑하는 함수입니다.
additionalResources	**CodeRef<WatchK8sResources<R>>**	제공됨	**mapper** 함수로 가져와 전달할 추가 리소스입니다.

console.dashboards/overview/prometheus/activity/resource
활동 트리거가 Kubernetes 리소스 감시를 기반으로 하는 Prometheus 개요 대시보드의 활동 카드에 활동을 추가합니다.

이름	값 유형	선택 사항	설명
queries	**string[]**	제공되지 않음	감시할 쿼리입니다.
component	**CodeRef<React.ComponentType<PrometheusActivityProps>>**	제공되지 않음	작업 구성 요소입니다.
isActivity	**CodeRef<(results: PrometheusResponse[]) ⇒ boolean>**	제공됨	지정된 리소스가 작업을 나타내는지 여부를 결정하는 함수입니다. 정의되지 않은 경우 모든 리소스는 활동을 나타냅니다.

console.dashboards/project/overview/item

프로젝트 개요 인벤토리 카드에 리소스 타일을 추가합니다.

이름	값 유형	선택 사항	설명
model	**CodeRef<T>**	제공되지 않음	가져올 **resource**의 모델입니다. 모델의 **label** 또는 **abbr**를 가져오는 데 사용됩니다.
mapper	**CodeRef<StatusGroupMapper<T, R>>**	제공됨	다양한 상태를 그룹에 매핑하는 함수입니다.
additionalResources	**CodeRef<WatchK8sResources<R>>**	제공됨	**mapper** 함수로 가져와 전달할 추가 리소스입니다.

console.dashboards/tab
개요 탭 뒤에 배치된 새 대시보드 탭을 추가합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	탭 링크 **href** 및 이 탭에 카드를 추가할 때 사용되는 고유한 탭 식별자입니다.
navSection	**'home' | 'storage'**	제공되지 않음	탭이 속한 탐색 섹션입니다.
title	**string**	제공되지 않음	탭의 이름입니다.

console.file-upload
이 확장자는 특정 파일 확장자에 파일 드롭 작업에 대한 처리기를 제공하는 데 사용할 수 있습니다.

이름	값 유형	선택 사항	설명
fileExtensions	**string[]**	제공되지 않음	지원되는 파일 확장자입니다.
handler	**CodeRef<FileUploadHandler>**	제공되지 않음	파일 드롭 작업을 처리하는 함수입니다.

console.flag
웹 콘솔 기능 플래그를 완전히 제어합니다.

이름	값 유형	선택 사항	설명
handler	**CodeRef<FeatureFlagHandler>**	제공되지 않음	임의의 기능 플래그를 설정하거나 설정 해제하는 데 사용됩니다.

console.flag/hookProvider
후크 처리기를 사용하여 웹 콘솔 기능 플래그를 완전히 제어할 수 있습니다.

이름	값 유형	선택 사항	설명
handler	**CodeRef<FeatureFlagHandler>**	제공되지 않음	임의의 기능 플래그를 설정하거나 설정 해제하는 데 사용됩니다.

console.flag/model
클러스터에서 CRD(CustomResourceDefinition) 오브젝트가 있어 새로운 웹 콘솔 기능 플래그를 추가합니다.

이름	값 유형	선택 사항	설명
flag	**string**	제공되지 않음	CRD가 감지된 후 설정할 플래그의 이름입니다.
model	**ExtensionK8sModel**	제공되지 않음	CRD를 참조하는 모델입니다.

console.global-config
이 확장은 클러스터 구성을 관리하는 데 사용되는 리소스를 식별합니다. 리소스에 대한 링크가 관리 → 클러스터 설정 → 구성 페이지에 추가됩니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	클러스터 구성 리소스 인스턴스의 고유 식별자입니다.
name	**string**	제공되지 않음	클러스터 구성 리소스 인스턴스의 이름입니다.
model	**ExtensionK8sModel**	제공되지 않음	클러스터 구성 리소스를 참조하는 모델입니다.
네임스페이스	**string**	제공되지 않음	클러스터 구성 리소스 인스턴스의 네임스페이스입니다.

console.model-metadata
API 검색을 통해 검색된 값을 재정의하여 모델의 표시를 사용자 지정합니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sGroupModel**	제공되지 않음	사용자 정의할 모델입니다. 그룹 또는 선택적 버전 및 종류만 지정할 수 있습니다.
badge	**ModelBadge**	제공됨	이 모델 참조를 기술 프리뷰 또는 개발자 프리뷰로 고려할지 여부입니다.
color	**string**	제공됨	이 모델에 연결할 색상입니다.
label	**string**	제공됨	라벨을 재정의합니다. **kind** 가 제공되어야 합니다.
labelPlural	**string**	제공됨	plural 라벨을 재정의합니다. **kind** 가 제공되어야 합니다.
abbr	**string**	제공됨	약어를 사용자 지정합니다. 기본값은 **kind**에서 모든 대문자로, 최대 4자 길이입니다. 이러한 **kind**가 제공되어야 합니다.

console.navigation/href
이 확장 기능을 사용하여 UI의 특정 링크를 가리키는 탐색 항목을 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 항목의 고유 식별자입니다.
name	**string**	제공되지 않음	이 항목의 이름입니다.
href	**string**	제공되지 않음	링크 **href** 값입니다.
perspective	**string**	제공됨	이 항목이 속한 화면 ID입니다. 지정하지 않으면 기본 화면이 설정됩니다.

이름	값 유형	선택 사항	설명
section	**string**	제공됨	이 항목이 속한 탐색 섹션입니다. 지정하지 않으면 이 항목을 최상위 링크로 렌더링합니다.
dataAttributes	**{ [key: string]: string; }**	제공됨	DOM에 데이터 속성을 추가합니다.
startsWith	**string[]**	제공됨	URL이 이러한 경로 중 하나로 시작될 때 이 항목을 활성으로 표시합니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 가 우선합니다.
namespaced	**boolean**	제공됨	**true** 인 경우 **/ns/active-namespace** 를 끝에 추가합니다.
prefixNamespaced	**boolean**	제공됨	**true** 인 경우 시작에 **/k8s/ns/active-namespace** 를 추가합니다.

console.navigation/resource-cluster
이 확장은 클러스터 리소스 세부 정보 페이지를 가리키는 탐색 항목을 제공하는 데 사용할 수 있습니다. 해당 리소스의 K8s 모델을 사용하여 탐색 항목을 정의할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 항목의 고유 식별자입니다.
model	**ExtensionK8sModel**	제공되지 않음	이 탐색 항목이 연결되는 모델입니다.

이름	값 유형	선택 사항	설명
perspective	**string**	제공됨	이 항목이 속한 화면 ID입니다. 지정하지 않으면 기본 화면이 설정됩니다.
section	**string**	제공됨	이 항목이 속한 탐색 섹션입니다. 지정하지 않으면 이 항목을 최상위 링크로 렌더링합니다.
dataAttributes	**{ [key: string]: string; }**	제공됨	DOM에 데이터 속성을 추가합니다.
startsWith	**string[]**	제공됨	URL이 이러한 경로 중 하나로 시작될 때 이 항목을 활성으로 표시합니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 가 우선합니다.
name	**string**	제공됨	기본 이름을 덮어씁니다. 연결 이름을 제공하지 않으면 모델의 복수 값과 동일합니다.

console.navigation/resource-ns
이 확장은 네임스페이스가 지정된 리소스 세부 정보 페이지를 가리키는 탐색 항목을 제공하는 데 사용할 수 있습니다. 해당 리소스의 K8s 모델을 사용하여 탐색 항목을 정의할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 항목의 고유 식별자입니다.
model	**ExtensionK8sModel**	제공되지 않음	이 탐색 항목이 연결되는 모델입니다.

이름	값 유형	선택 사항	설명
perspective	**string**	제공됨	이 항목이 속한 화면 ID입니다. 지정하지 않으면 기본 화면이 설정됩니다.
section	**string**	제공됨	이 항목이 속한 탐색 섹션입니다. 지정하지 않으면 이 항목을 최상위 링크로 렌더링합니다.
dataAttributes	**{ [key: string]: string; }**	제공됨	DOM에 데이터 속성을 추가합니다.
startsWith	**string[]**	제공됨	URL이 이러한 경로 중 하나로 시작될 때 이 항목을 활성으로 표시합니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 가 우선합니다.
name	**string**	제공됨	기본 이름을 덮어씁니다. 연결 이름을 제공하지 않으면 모델의 복수 값과 동일합니다.

console.navigation/section
이 확장은 탐색 탭에서 탐색 항목의 새 섹션을 정의하는 데 사용할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 항목의 고유 식별자입니다.
perspective	**string**	제공됨	이 항목이 속한 화면 ID입니다. 지정하지 않으면 기본 화면이 설정됩니다.
dataAttributes	**{ [key: string]: string; }**	제공됨	DOM에 데이터 속성을 추가합니다.

이름	값 유형	선택 사항	설명
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 가 우선합니다.
name	**string**	제공됨	이 섹션의 이름입니다. 지정하지 않으면 섹션 위에 구분자만 표시됩니다.

console.navigation/separator
이 확장은 탐색의 탐색 항목 사이에 구분자를 추가하는 데 사용할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 항목의 고유 식별자입니다.
perspective	**string**	제공됨	이 항목이 속한 화면 ID입니다. 지정하지 않으면 기본 화면이 설정됩니다.
section	**string**	제공됨	이 항목이 속한 탐색 섹션입니다. 지정하지 않으면 이 항목을 최상위 링크로 렌더링합니다.
dataAttributes	**{ [key: string]: string; }**	제공됨	DOM에 데이터 속성을 추가합니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 가 우선합니다.

console.page/resource/details

이름	값 유형	선택 사항	설명
model	**ExtensionK8sGroupKindModel**	제공되지 않음	이 리소스 페이지가 연결되는 모델입니다.
component	**CodeRef<React.ComponentType<{ match: match<{}>; namespace: string; model: ExtensionK8sModel; }>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.

console.page/resource/list
콘솔 라우터에 새 리소스 목록 페이지를 추가합니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sGroupKindModel**	제공되지 않음	이 리소스 페이지가 연결되는 모델입니다.
component	**CodeRef<React.ComponentType<{ match: match<{}>; namespace: string; model: ExtensionK8sModel; }>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.

console.page/route
웹 콘솔 라우터에 새 페이지를 추가합니다. React Router 를 참조하십시오.

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<RouteComponentProps<{}, StaticContext, any>>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.
path	**string | string[]**	제공되지 않음	**path-to-regexp@^1.7.0** 에서 이해할 수 있는 유효한 URL 경로 또는 경로 배열입니다.

이름	값 유형	선택 사항	설명
perspective	**string**	제공됨	이 페이지가 속한 화면입니다. 지정하지 않으면 모든 화면이 설정됩니다.
exact	**boolean**	제공됨	true인 경우 경로는 **location.pathname** 과 정확히 일치하는 경우에만 일치합니다.

console.page/route/standalone
웹 콘솔 라우터에 일반 페이지 레이아웃 외부에서 렌더링된 새 독립 실행형 페이지를 웹 콘솔 라우터에 추가합니다. React Router **를 참조하십시오.**

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<RouteComponentProps<{}, StaticContext, any>>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.
path	**string | string[]**	제공되지 않음	**path-to-regexp@^1.7.0** 에서 이해할 수 있는 유효한 URL 경로 또는 경로 배열입니다.
exact	**boolean**	제공됨	true인 경우 경로는 **location.pathname** 과 정확히 일치하는 경우에만 일치합니다.

console.perspective
이 확장 기능은 콘솔에 새로운 화면을 제공하여 탐색 메뉴를 사용자 지정할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	화면 식별자입니다.
name	**string**	제공되지 않음	화면 표시 이름입니다.
icon	**CodeRef<LazyComponent>**	제공되지 않음	화면 표시 아이콘입니다.

이름	값 유형	선택 사항	설명
landingPageURL	**CodeRef<(flags: { [key: string]: boolean; }, isFirstVisit: boolean) ⇒ string>**	제공되지 않음	화면 검색 페이지 URL을 가져오는 기능입니다.
importRedirectURL	**CodeRef<(namespace: string) ⇒ string>**	제공되지 않음	가져오기 흐름을 위한 리디렉션 URL을 가져오는 함수입니다.
default	**boolean**	제공됨	화면이 기본값인지 여부입니다. 기본값은 하나만 있을 수 있습니다.
defaultPins	**ExtensionK8sModel[]**	제공됨	nav의 기본 고정 리소스
usePerspectiveDetection	**CodeRef<() ⇒ [boolean, boolean]>**	제공됨	기본 화면을 감지하는 후크

console.project-overview/inventory-item
프로젝트 개요 페이지에 새 인벤토리 항목을 추가합니다.

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<{ projectName: string; }>>**	제공되지 않음	렌더링할 구성 요소입니다.

console.project-overview/utilization-item
새 프로젝트 개요 사용률 항목을 추가합니다.

이름	값 유형	선택 사항	설명
title	**string**	제공되지 않음	사용률 항목의 제목입니다.
getUtilizationQuery	**CodeRef<GetProjectQuery>**	제공되지 않음	Prometheus 사용률 쿼리.
humanize	**CodeRef<Humanize>**	제공되지 않음	Prometheus 데이터를 사람이 읽을 수 있는 형식으로 변환합니다.

이름	값 유형	선택 사항	설명
getTotalQuery	**CodeRef<GetProjectQuery>**	제공됨	Prometheus 총 쿼리.
getRequestQuery	**CodeRef<GetProjectQuery>**	제공됨	Prometheus 요청 쿼리.
getLimitQuery	**CodeRef<GetProjectQuery>**	제공됨	Prometheus 제한 쿼리.
TopConsumerPopover	**CodeRef<React.ComponentType<TopConsumerPopoverProps>>**	제공됨	일반 값 대신 상위 소비자 팝업을 표시합니다.

console.pvc/alert
이 확장 기능을 사용하여 PVC 세부 정보 페이지에서 사용자 정의 경고를 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
alert	**CodeRef<React.ComponentType<{ pvc: K8sResourceCommon; }>>**	제공되지 않음	경고 구성 요소입니다.

console.pvc/create-prop
이 확장 기능을 사용하면 PVC 목록 페이지에서 PVC 리소스를 생성할 때 사용할 추가 속성을 지정할 수 있습니다.

이름	값 유형	선택 사항	설명
label	**string**	제공되지 않음	prop 만들기 작업의 레이블입니다.
path	**string**	제공되지 않음	prop 작업 만들기의 경로입니다.

console.pvc/delete
이 확장을 사용하면 PVC 리소스를 삭제할 수 있습니다. 추가 정보 및 사용자 정의 PVC 삭제 논리가 포함된 경고를 제공할 수 있습니다.

이름	값 유형	선택 사항	설명

이름	값 유형	선택 사항	설명
predicate	**CodeRef<(pvc: K8sResourceCommon) ⇒ boolean>**	제공되지 않음	확장을 사용할지 여부를 알려주는 서술자입니다.
onPVCKill	**CodeRef<(pvc: K8sResourceCommon) ⇒ Promise<void>>**	제공되지 않음	PVC 삭제 작업 방법입니다.
alert	**CodeRef<React.ComponentType<{ pvc: K8sResourceCommon; }>>**	제공되지 않음	추가 정보를 표시하는 경고 구성 요소입니다.

console.pvc/status

이름	값 유형	선택 사항	설명
priority	**number**	제공되지 않음	상태 구성 요소의 우선 순위입니다. 값이 클수록 우선순위가 높습니다.
status	**CodeRef<React.ComponentType<{ pvc: K8sResourceCommon; }>>**	제공되지 않음	상태 구성 요소입니다.
predicate	**CodeRef<(pvc: K8sResourceCommon) ⇒ boolean>**	제공되지 않음	상태 구성 요소를 렌더링할지 여부를 나타내는 서술자입니다.

console.redux-reducer
plugins.<scope> 하위 상태에서 작동하는 Console Redux 저장소에 새 축소기를 추가합니다.

이름	값 유형	선택 사항	설명
scope	**string**	제공되지 않음	Redux 상태 오브젝트 내에서 reducer-managed 하위 상태를 나타내는 키입니다.
reducer	**CodeRef<Reducer<any, AnyAction>>**	제공되지 않음	축소기 기능으로, reducer-managed 하위 상태로 작동합니다.

console.resource/create
이 확장을 사용하면 사용자가 새 리소스 인스턴스를 만들려고 할 때 렌더링되는 특정 리소스에 대한 사용자 지정 구성 요소(예: 마법사 또는 양식)를 플러그인에 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sModel**	제공되지 않음	이 리소스 페이지가 렌더링되는 모델
component	**CodeRef<React.ComponentType<CreateResourceComponentProps>>**	제공되지 않음	모델이 일치하면 렌더링할 구성 요소입니다.

console.resource/details-item
세부 정보 페이지의 기본 리소스 요약에 새 세부 정보 항목을 추가합니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sModel**	제공되지 않음	subject 리소스의 API 그룹, 버전 및 종류입니다.
id	**string**	제공되지 않음	고유 식별자입니다.
열	**DetailsItemColumn**	제공되지 않음	세부 정보 페이지의 리소스 요약의 '왼쪽' 또는 '오른쪽' 열에 항목이 표시되는지 확인합니다. 기본값: 'right'
title	**string**	제공되지 않음	세부 정보 항목 제목입니다.
path	**string**	제공됨	details 항목 값으로 사용되는 리소스 속성의 선택적 정규화된 경로입니다. 기본 유형 값만 직접 렌더링할 수 있습니다. 구성 요소 속성을 사용하여 다른 데이터 유형을 처리합니다.
component	**CodeRef<React.ComponentType<DetailsItemComponentProps<K8sResourceCommon, any>>>**	제공됨	세부 정보 항목 값을 렌더링하는 선택적 React 구성 요소입니다.

이름	값 유형	선택 사항	설명
sortWeight	**number**	제공됨	동일한 열에 있는 다른 모든 세부 정보 항목을 기준으로 하는 선택적 정렬 가중치입니다. 유효한 JavaScriptNumber 로 표시됩니다. 각 열의 항목은 독립적으로 정렬되며 가장 낮은 항목이 가장 높습니다. 정렬 가중치가 없는 항목은 정렬 가중치가 있는 항목 다음에 정렬됩니다.

console.storage-class/provisioner
스토리지 클래스를 생성하는 동안 옵션으로 새 스토리지 클래스 프로비전 프로그램을 추가합니다.

이름	값 유형	선택 사항	설명
CSI	**ProvisionerDetails**	제공됨	컨테이너 스토리지 인터페이스 프로비저너 유형
OTHERS	**ProvisionerDetails**	제공됨	기타 프로비저너 유형

console.storage-provider
이 확장을 사용하여 스토리지 및 공급자별 구성 요소를 연결할 때 선택하는 새 스토리지 공급자를 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
name	**string**	제공되지 않음	공급자의 표시 이름입니다.
Component	**CodeRef<React.ComponentType<Partial<RouteComponentProps<{}, StaticContext, any>>>>**	제공되지 않음	렌더링할 공급자별 구성 요소입니다.

console.tab
contextId 와 일치하는 수평 nav에 탭을 추가합니다.

이름	값 유형	선택 사항	설명

이름	값 유형	선택 사항	설명
contextId	**string**	제공되지 않음	탭을 삽입할 수평 nav에 할당된 컨텍스트 ID입니다. 가능한 값: **dev-console-observe**
name	**string**	제공되지 않음	탭의 표시 라벨
href	**string**	제공되지 않음	기존 URL에 추가된 **href**
component	**CodeRef<React.ComponentType<PageComponentProps<K8sResourceCommon>>>**	제공되지 않음	탭 콘텐츠 구성 요소.

console.tab/horizontalNav
이 확장 기능을 사용하여 리소스 세부 정보 페이지에 탭을 추가할 수 있습니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sKindVersionModel**	제공되지 않음	이 공급자가 보여주는 모델입니다.
page	**{ name: string; href: string; }**	제공되지 않음	수평 탭에 표시할 페이지입니다. 탭 이름을 이름으로 사용하고 탭의 href를 사용합니다.
component	**CodeRef<React.ComponentType<PageComponentProps<K8sResourceCommon>>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.

console.telemetry/listener
이 구성 요소는 Telemetry 이벤트를 수신하는 리스너 기능을 등록하는 데 사용할 수 있습니다. 이러한 이벤트에는 사용자 ID, 페이지 탐색 및 기타 애플리케이션별 이벤트가 포함됩니다. 리스너는 이 데이터를 보고 및 분석 목적으로 사용할 수 있습니다.

이름	값 유형	선택 사항	설명
listener	**CodeRef<TelemetryEventListener>**	제공되지 않음	Telemetry 이벤트 수신

console.topology/adapter/build
BuildAdapter 는 빌드 구성 요소에서 사용할 수 있는 데이터에 요소를 조정하기 위해 어댑터를 기여합니다.

이름	값 유형	선택 사항	설명
adapt	**CodeRef<(element: GraphElement) ⇒ AdapterDataType<BuildConfigData> | undefined>**	제공되지 않음	Build 구성 요소에서 사용할 수 있는 데이터에 대한 요소를 조정하기 위한 어댑터입니다.

console.topology/adapter/network
NetworkAdapater 는 어댑터를 기여하여 Networking 구성 요소에서 사용할 수 있는 데이터에 요소를 조정합니다.

이름	값 유형	선택 사항	설명
adapt	**CodeRef<(element: GraphElement) ⇒ NetworkAdapterType | undefined>**	제공되지 않음	네트워킹 구성 요소에서 사용할 수 있는 데이터에 대한 요소를 조정하기 위한 어댑터입니다.

console.topology/adapter/pod
PodAdapter 는 Pod 구성 요소에서 사용할 수 있는 데이터에 요소를 조정하는 어댑터를 제공합니다.

이름	값 유형	선택 사항	설명
adapt	**CodeRef<(element: GraphElement) ⇒ AdapterDataType<PodsAdapterDataType> | undefined>**	제공되지 않음	Adapter는 Pod 구성 요소에서 사용할 수 있는 데이터에 맞게 요소를 조정합니다.

console.topology/component/factory
Getter for a **ViewComponentFactory**.

이름	값 유형	선택 사항	설명
getFactory	**CodeRef<ViewComponentFactory>**	제공되지 않음	**ViewComponentFactory**에 대한 Getter입니다.

console.topology/create/connector
생성 커넥터 함수에 대한 getter입니다.

이름	값 유형	선택 사항	설명
getCreateConnector	**CodeRef<CreateConnectionGetter>**	제공되지 않음	생성 커넥터 함수에 대한 getter입니다.

console.topology/data/factory
토폴로지 데이터 모델 팩토리 확장

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	팩토리의 고유 ID입니다.
priority	**number**	제공되지 않음	팩토리의 우선 순위
resources	**WatchK8sResources Generic**	제공됨	**useK8sWatchResources** 후크에서 가져올 리소스입니다.
workloadKeys	**string[]**	제공됨	워크로드가 포함된 리소스의 키입니다.
getDataModel	**CodeRef<TopologyDataModelGetter>**	제공됨	데이터 모델 팩토리에 대한 getter입니다.
isResourceDepicted	**CodeRef<TopologyDataModelDepicted>**	제공됨	이 모델 팩토리에 의해 리소스가 표시되어 있는지 확인하는 함수에 대한 getter입니다.
getDataModelReconciler	**CodeRef<TopologyDataModelReconciler>**	제공됨	모든 확장 모델의 모델이 로드된 후 데이터 모델을 조정하는 기능에 대한 getter입니다.

console.topology/decorator/provider
토폴로지 데코레이터 공급자 확장

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	확장과 관련된 토폴로지 데코레이터의 ID
priority	**number**	제공되지 않음	확장과 관련된 토폴로지 데코레이터의 우선 순위
quadrant	**TopologyQuadrant**	제공되지 않음	확장과 관련된 토폴로지 데코레이터에 대한 Quadrant
decorator	**CodeRef<TopologyDecoratorGetter>**	제공되지 않음	확장과 관련된 데코레이터

console.topology/details/resource-alert
DetailsResourceAlert 는 특정 토폴로지 컨텍스트 또는 graph 요소에 대한 경고를 제공합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 경고의 ID입니다. 경고가 표시된 후 경고를 표시하지 않아야 하는 경우 상태를 저장하는 데 사용됩니다.
contentProvider	**CodeRef<(element: GraphElement) ⇒ DetailsResourceAlertContent | null>**	제공되지 않음	경고 내용을 반환하는 후크입니다.

console.topology/details/resource-link
DetailsResourceLink 특정 토폴로지 컨텍스트 또는 그래프 요소에 대한 링크를 기여합니다.

이름	값 유형	선택 사항	설명
link	**CodeRef<(element: GraphElement) ⇒ React.Component | undefined>**	제공되지 않음	제공된 경우 리소스 링크를 반환하고, 그러지 않으면 정의되지 않습니다. 스타일에 **ResourceIcon** 및 **ResourceLink** 속성을 사용합니다.
priority	**number**	제공됨	높은 우선 순위 팩토리에서 링크를 만들 수있는 첫 번째 기회를 얻습니다.

console.topology/details/tab
DetailsTab 은 토폴로지 세부 정보 패널의 탭을 제공합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 세부 정보 탭의 고유 식별자입니다.
label	**string**	제공되지 않음	UI에 표시할 탭 레이블입니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.

이름	값 유형	선택 사항	설명
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 값이 우선합니다.

console.topology/details/tab-section
DetailsTabSection 은 토폴로지 세부 정보 패널의 특정 탭에 대한 섹션을 제공합니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	이 세부 정보 탭 섹션의 고유 식별자입니다.
tab	**string**	제공되지 않음	이 섹션이 제공해야 하는 상위 탭 ID입니다.
provider	**CodeRef<DetailsTabSectionExtensionHook>**	제공되지 않음	구성 요소를 반환하는 후크 또는 null 또는 정의되지 않은 경우 토폴로지 사이드바에 렌더링됩니다. SDK 구성 요소: **<Section title=\{}>…** 패딩 영역
section	**CodeRef<(element: GraphElement, renderNull?: () ⇒ null) ⇒ React.Component | undefined>**	제공되지 않음	더 이상 사용되지 않음: 공급자가 정의되지 않은 경우 Fallback. renderNull은 이미 작동하지 않습니다.
insertBefore	**string | string[]**	제공됨	여기서 참조하는 항목 앞에 이 항목을 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다.
insertAfter	**string | string[]**	제공됨	이 항목을 여기에 참조한 항목 뒤에 삽입합니다. 배열의 경우 첫 번째 항목이 순서대로 사용됩니다. **insertBefore** 값이 우선합니다.

console.topology/display/filters

토폴로지 표시 필터 확장

이름	값 유형	선택 사항	설명
getTopologyFilters	**CodeRef<() ⇒ TopologyDisplayOption[]>**	제공되지 않음	확장과 관련된 토폴로지 필터의 getter
applyDisplayOptions	**CodeRef<TopologyApplyDisplayOptions>**	제공되지 않음	모델에 필터를 적용하는 함수

console.topology/relationship/provider
토폴로지 관계 공급자 커넥터 확장

이름	값 유형	선택 사항	설명
provides	**CodeRef<RelationshipProviderProvides>**	제공되지 않음	소스 노드와 대상 노드 사이에 연결을 생성할 수 있는지 확인하는 데 사용됩니다.
tooltip	**string**	제공되지 않음	툴팁은 커넥터 작업이 드롭 대상 위에 마우스를 가져가는 시기를 표시합니다(예: "Visual Connector 만들기")
create	**CodeRef<RelationshipProviderCreate>**	제공되지 않음	연결을 생성하기 위해 커넥터가 대상 노드 위에 드롭될 때 실행되는 콜백
priority	**number**	제공되지 않음	관계의 우선 순위는 여러 개일 경우 더 높은 것이 우선됩니다.

console.user-preference/group
이 확장은 콘솔 사용자 참조 페이지에 그룹을 추가하는 데 사용할 수 있습니다. 콘솔 사용자 참조 페이지에 세로 탭 옵션으로 표시됩니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	사용자 기본 설정 그룹을 식별하는 데 사용되는 ID입니다.
label	**string**	제공되지 않음	사용자 기본 설정 그룹의 레이블

이름	값 유형	선택 사항	설명
insertBefore	**string**	제공됨	이 그룹을 배치하기 전에 사용자 기본 설정 그룹의 ID
insertAfter	**string**	제공됨	이 그룹을 배치해야 하는 사용자 기본 그룹의 ID

console.user-preference/item
이 확장 기능을 사용하여 콘솔 사용자 기본 설정 페이지의 사용자 기본 설정 그룹에 항목을 추가할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	사용자 기본 항목을 식별하고 insertAfter에서 참조되고 항목 순서를 정의하기 전에 삽입하는 데 사용되는 ID
label	**string**	제공되지 않음	사용자 기본 설정의 레이블
description	**string**	제공되지 않음	사용자 우선순위에 대한 설명
field	**UserPreferenceField**	제공되지 않음	사용자 기본 설정을 위해 값을 렌더링하는 데 사용되는 입력 필드 옵션
groupId	**string**	제공됨	항목이 속한 사용자 기본 설정 그룹을 식별하는 데 사용되는 ID
insertBefore	**string**	제공됨	이 항목을 배치하기 전에 사용자 기본 설정 항목의 ID
insertAfter	**string**	제공됨	이 항목을 배치해야 하는 사용자 기본 설정 항목의 ID

console.yaml-template
yaml 편집기를 통해 리소스를 편집하기 위한 YAML 템플릿입니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sModel**	제공되지 않음	템플릿과 관련된 모델입니다.
템플릿	**CodeRef<string>**	제공되지 않음	YAML 템플릿입니다.
name	**string**	제공되지 않음	템플릿의 이름입니다. 이를 기본 템플릿으로 표시하려면 **default** 이름을 사용합니다.

dev-console.add/action
이 확장 기능을 통해 플러그인은 개발자 화면의 추가 페이지에 작업 항목을 추가할 수 있습니다. 예를 들어 서버리스 플러그인은 개발자 콘솔의 추가 페이지에 서버리스 함수를 추가하기 위한 새 작업 항목을 추가할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	작업을 식별하는 데 사용되는 ID입니다.
label	**string**	제공되지 않음	작업의 레이블입니다.
description	**string**	제공되지 않음	작업에 대한 설명입니다.
href	**string**	제공되지 않음	탐색할 **href** 입니다.
groupId	**string**	제공됨	작업이 속한 작업 그룹을 식별하는 데 사용되는 ID입니다.
icon	**CodeRef<React.ReactNode>**	제공됨	화면 표시 아이콘입니다.
accessReview	**AccessReviewResourceAttributes[]**	제공됨	작업의 가시성 또는 사용을 제어하는 선택적 액세스 검토입니다.

dev-console.add/action-group
이 확장 기능을 사용하면 플러그인에서 개발자 콘솔의 추가 페이지에 그룹을 구성할 수 있습니다. 확장 정의에 따라 추가 작업 페이지에서 함께 그룹화되는 작업으로 그룹을 참조할 수 있습니다. 예를 들어 서버리스 플러그인은 서버리스 그룹에 여러 추가 작업과 함께 제공할 수 있습니다.

이름	값 유형	선택 사항	설명
id	**string**	제공되지 않음	작업 그룹을 식별하는 데 사용되는 ID

이름	값 유형	선택 사항	설명
name	**string**	제공되지 않음	작업 그룹의 제목
insertBefore	**string**	제공됨	이 그룹을 배치하기 전에 작업 그룹의 ID
insertAfter	**string**	제공됨	이 그룹을 배치해야 하는 작업 그룹의 ID

dev-console.import/environment
이 확장 기능을 사용하여 개발자 콘솔 Git 가져오기 양식의 빌더 이미지 선택기 아래에 추가 빌드 환경 변수 필드를 지정할 수 있습니다. 설정하면 필드가 빌드 섹션에서 동일한 이름의 환경 변수를 재정의합니다.

이름	값 유형	선택 사항	설명
imageStreamName	**string**	제공되지 않음	사용자 정의 환경 변수를 제공하는 이미지 스트림 이름
imageStreamTags	**string[]**	제공되지 않음	지원되는 이미지 스트림 태그 목록
environments	**ImageEnvironment[]**	제공되지 않음	환경 변수 목록

console.dashboards/overview/detail/item
더 이상 사용되지 않습니다. 대신 CustomOverviewDetailItem 유형을 사용합니다.

이름	값 유형	선택 사항	설명
component	**CodeRef<React.ComponentType<{}>>**	제공되지 않음	**DetailItem** 구성 요소를 기반으로 하는 값

console.page/resource/tab
더 이상 사용되지 않습니다. 대신 console.tab/horizontalNav 를 사용합니다. 콘솔 라우터에 새 리소스 탭 페이지를 추가합니다.

이름	값 유형	선택 사항	설명
model	**ExtensionK8sGroupKindModel**	제공되지 않음	이 리소스 페이지가 연결되는 모델입니다.
component	**CodeRef<React.ComponentType<RouteComponentProps<{}, StaticContext, any>>>**	제공되지 않음	경로가 일치하면 렌더링할 구성 요소입니다.

이름	값 유형	선택 사항	설명
name	**string**	제공되지 않음	탭의 이름입니다.
href	**string**	제공됨	탭 링크의 **href** 선택 사항입니다. 지정하지 않으면 첫 번째 **path**가 사용됩니다.
exact	**boolean**	제공됨	true인 경우 경로는 **location.pathname** 과 정확히 일치하는 경우에만 일치합니다.

4.5.2. 동적 플러그인 API

useActivePerspective
현재 활성화된 화면과 활성 화면을 설정하기 위한 콜백을 제공하는 후크입니다. 현재 활성 화면 및 setter 콜백을 포함하는 튜플을 반환합니다.

예제

```
const Component: React.FC = (props) => {
  const [activePerspective, setActivePerspective] = useActivePerspective();
  return <select
    value={activePerspective}
    onChange={(e) => setActivePerspective(e.target.value)}
  >
    {
      // ...perspective options
    }
  </select>
}
```

GreenCheckCircleIcon
녹색 체크 표시 원 아이콘을 표시하기 위한 구성 요소입니다.

예제

```
<GreenCheckCircleIcon title="Healthy" />
```

매개변수 이름	설명
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
title	(선택 사항) 아이콘 제목
size	(선택 사항) 아이콘 크기: (**sm**,**md**,**lg**,**xl**)

RedExclamationCircleIcon
빨간색 느낌표 원 아이콘을 표시하는 구성 요소입니다.

예제

```
<RedExclamationCircleIcon title="Failed" />
```

매개변수 이름	설명
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
title	(선택 사항) 아이콘 제목
size	(선택 사항) 아이콘 크기: (**sm**,**md**,**lg**,**xl**)

YellowExclamationTriangleIcon
노란색 구분 기호 아이콘을 표시하는 구성 요소입니다.

예제

```
<YellowExclamationTriangleIcon title="Warning" />
```

매개변수 이름	설명
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
title	(선택 사항) 아이콘 제목
size	(선택 사항) 아이콘 크기: (**sm**,**md**,**lg**,**xl**)

BlueInfoCircleIcon
파란색 정보 원 아이콘을 표시하는 구성 요소입니다.

예제

```
<BlueInfoCircleIcon title="Info" />
```

매개변수 이름	설명
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
title	(선택 사항) 아이콘 제목
size	(선택 사항) 아이콘 크기: ('sm', 'md', 'lg', 'xl')

ErrorStatus

오류 상태 팝업을 표시하는 구성 요소입니다.

예제

```
<ErrorStatus title={errorMsg} />
```

매개변수 이름	설명
title	(선택 사항) 상태 텍스트
iconOnly	(선택 사항) true인 경우 아이콘만 표시합니다.
noTooltip	(선택 사항) true인 경우 툴팁이 표시되지 않습니다.
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
popoverTitle	(선택 사항) 팝업의 제목

InfoStatus
정보 상태 팝업을 표시하는 구성 요소입니다.

예제

```
<InfoStatus title={infoMsg} />
```

매개변수 이름	설명
title	(선택 사항) 상태 텍스트
iconOnly	(선택 사항) true인 경우 아이콘만 표시합니다.
noTooltip	(선택 사항) true인 경우 툴팁이 표시되지 않습니다.
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
popoverTitle	(선택 사항) 팝업의 제목

ProgressStatus
진행 상태 팝업을 표시하는 구성 요소입니다.

예제

```
<ProgressStatus title={progressMsg} />
```

매개변수 이름	설명
title	(선택 사항) 상태 텍스트
iconOnly	(선택 사항) true인 경우 아이콘만 표시합니다.
noTooltip	(선택 사항) true인 경우 툴팁이 표시되지 않습니다.
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
popoverTitle	(선택 사항) 팝업의 제목

successStatus
성공 상태 팝업을 표시하는 구성 요소입니다.

예제

```
<SuccessStatus title={successMsg} />
```

매개변수 이름	설명
title	(선택 사항) 상태 텍스트
iconOnly	(선택 사항) true인 경우 아이콘만 표시합니다.
noTooltip	(선택 사항) true인 경우 툴팁이 표시되지 않습니다.
className	(선택 사항) 구성 요소의 추가 클래스 이름입니다.
popoverTitle	(선택 사항) 팝업의 제목

checkAccess
지정된 리소스에 대한 사용자 액세스에 대한 정보를 제공합니다. 리소스 액세스 정보를 사용하여 오브젝트를 반환합니다.

매개변수 이름	설명
resourceAttributes	액세스 검토를 위한 리소스 속성
impersonate	명의 도용 세부 정보

useAccessReview
지정된 리소스에 대한 사용자 액세스 정보를 제공하는 후크입니다. isAllowed 및 loading 값을 사용하여 배열을 반환합니다.

매개변수 이름	설명
resourceAttributes	액세스 검토를 위한 리소스 속성
impersonate	명의 도용 세부 정보

useResolvedExtensions
확인된 CodeRef 속성과 함께 콘솔 확장을 사용하기 위한 React 후크입니다. 이 후크는useExtensions 후크로 동일한 인수를 수락하고 조정된 확장 인스턴스 목록을 반환하여 각 확장 기능 속성 내의 모든 코드 참조를 해결합니다.

처음에 후크는 빈 배열을 반환합니다. 해결이 완료되면 후크가 적용된 확장 목록을 반환하여 React 구성 요소가 다시 렌더링됩니다. 일치하는 확장 목록이 변경되면 확인이 다시 시작됩니다. 후크는 확인이 완료될 때까지 이전 결과를 계속 반환합니다.

후크의 결과 요소는 재렌더링을 통해 참조로 안정적으로 유지됩니다. 해결된 코드 참조가 포함된 조정된 확장 인스턴스 목록, 해결이 완료되었는지 여부를 나타내는 부울 플래그, 해결 중에 감지된 오류 목록이 포함된 튜플을 반환합니다.

예제

```
const [navItemExtensions, navItemsResolved] = useResolvedExtensions<NavItem>
(isNavItem);
// process adapted extensions and render your component
```

매개변수 이름	설명
typeGuards	각각 동적 플러그인 확장을 인수로 수락하고 확장이 원하는 유형 제약 조건을 충족하는지 여부를 나타내는 부울 플래그를 반환하는 콜백 목록입니다.

HorizontalNav
페이지에 대한 탐색 모음을 생성하는 구성 요소입니다. 라우팅은 구성 요소의 일부로 처리됩니다. console.tab/horizontalNav 를 사용하여 모든 수평 탐색에 콘텐츠를 추가할 수 있습니다.

예제

```
const HomePage: React.FC = (props) => {
  const page = {
    href: '/home',
    name: 'Home',
    component: () => <>Home</>
  }
  return <HorizontalNav match={props.match} pages={[page]} />
}
```

매개변수 이름	설명
resource	K8sResourceCommon 유형의 오브젝트인 이 Navigation과 연결된 리소스
pages	페이지 오브젝트의 배열
match	React Router에서 제공하는 일치 오브젝트

VirtualizedTable
가상화된 테이블을 만드는 구성 요소입니다.

예제

```
const MachineList: React.FC<MachineListProps> = (props) => {
  return (
    <VirtualizedTable<MachineKind>
     {...props}
     aria-label='Machines'
     columns={getMachineColumns}
     Row={getMachineTableRow}
    />
  );
}
```

매개변수 이름	설명
data	데이터 표
loaded	데이터가 로드되었음을 나타내는 플래그
loadError	데이터 로드에 문제가 있는 경우 오류 개체
columns	열 설정
Row	행 설정
unfilteredData	필터가 없는 원본 데이터
NoDataEmptyMsg	(선택 사항) 데이터 빈 메시지 구성 요소가 없음
EmptyMsg	(선택 사항) 빈 메시지 구성 요소
scrollNode	(선택 사항) 스크롤을 처리하는 기능
label	(선택 사항) 표의 라벨

매개변수 이름	설명
ariaLabel	(선택 사항) aria 레이블
gridBreakPoint	응답성을 위해 그리드를 분할하는 방법의 크기 조정
onSelect	(선택 사항) 표 선택 처리를 위한 기능
rowData	(선택 사항) 행과 관련된 데이터

TableData
테이블 행 내에 테이블 데이터를 표시하는 구성 요소입니다.

예제

```
const PodRow: React.FC<RowProps<K8sResourceCommon>> = ({ obj, activeColumnIDs }) =>
{
  return (
    <>
      <TableData id={columns[0].id} activeColumnIDs={activeColumnIDs}>
        <ResourceLink kind="Pod" name={obj.metadata.name} namespace=
{obj.metadata.namespace} />
      </TableData>
      <TableData id={columns[1].id} activeColumnIDs={activeColumnIDs}>
        <ResourceLink kind="Namespace" name={obj.metadata.namespace} />
      </TableData>
    </>
  );
};
```

매개변수 이름	설명
id	테이블의 고유 ID
activeColumnIDs	활성 열
className	(선택 사항) 스타일링의 옵션 클래스 이름입니다.

useActiveColumns
사용자가 선택한 활성 TableColumns 목록을 제공하는 후크입니다.

예제

```
// See implementation for more details on TableColumn type
  const [activeColumns, userSettingsLoaded] = useActiveColumns({
    columns,
    showNamespaceOverride: false,
    columnManagementID,
```

```
  });
  return userSettingsAreLoaded ? <VirtualizedTable columns={activeColumns} {...otherProps}
/> : null
```

매개변수 이름	설명
options	키-값 맵으로 전달되는 방법
\{TableColumn[]} options.columns	사용 가능한 모든 TableColumns 배열
{boolean} [options.showNamespaceOverride]	(선택 사항) true인 경우 열 관리 선택 사항에 관계없이 네임스페이스 열이 포함됩니다.
{string} [options.columnManagementID]	(선택 사항) 사용자 설정에 열 관리 선택을 유지하고 검색하는 데 사용되는 고유 ID입니다. 일반적으로 리소스의 GVK(그룹/버전/종류) 문자열입니다.

현재 사용자가 선택한 활성 열(options.columns의 하위 집합)과 사용자 설정이 로드되었는지 여부를 나타내는 부울 플래그가 포함됩니다.

ListPageHeader
페이지 헤더를 생성하는 구성 요소입니다.

예제

```
const exampleList: React.FC = () => {
  return (
    <>
      <ListPageHeader title="Example List Page"/>
    </>
  );
};
```

매개변수 이름	설명
title	제목
helpText	(선택 사항) 반응 노드로서의 도움말 섹션
badge	(선택 사항) 반응 노드로 배지 아이콘

ListPageCreate
이 리소스의 YAML 생성에 대한 링크를 자동으로 생성하는 특정 리소스 종류에 대한 생성 버튼을 추가하는 구성 요소입니다.

예제

```
const exampleList: React.FC<MyProps> = () => {
  return (
```

```
    <>
      <ListPageHeader title="Example Pod List Page"/>
        <ListPageCreate groupVersionKind="Pod">Create Pod</ListPageCreate>
      </ListPageHeader>
    </>
  );
};
```

매개변수 이름	설명
groupVersionKind	표시할 리소스 그룹/버전/종류

ListPageCreateLink
제대로된 링크를 만들기 위한 구성 요소입니다.

예제

```
const exampleList: React.FC<MyProps> = () => {
 return (
  <>
   <ListPageHeader title="Example Pod List Page"/>
     <ListPageCreateLink to={'/link/to/my/page'}>Create Item</ListPageCreateLink>
   </ListPageHeader>
  </>
 );
};
```

매개변수 이름	설명
다음으로 변경	링크가 연결되어야 하는 문자열 위치
createAccessReview	(선택 사항) 액세스 확인에 사용되는 네임스페이스 및 종류가 있는 오브젝트
children	(선택 사항) 구성 요소에 대한 하위 항목

ListPageCreateButton
버튼을 생성하는 구성 요소입니다.

예제

```
const exampleList: React.FC<MyProps> = () => {
  return (
    <>
      <ListPageHeader title="Example Pod List Page"/>
        <ListPageCreateButton createAccessReview={access}>Create Pod</ListPageCreateButton>
      </ListPageHeader>
    </>
  );
};
```

매개변수 이름	설명
createAccessReview	(선택 사항) 액세스 확인에 사용되는 네임스페이스 및 종류가 있는 오브젝트
pfButtonProps	(선택 사항) Patternfly 버튼 props

ListPageCreateDropdown
권한 확인으로 래핑된 드롭다운을 생성하기 위한 구성 요소입니다.

예제

```
const exampleList: React.FC<MyProps> = () => {
  const items = {
    SAVE: 'Save',
    DELETE: 'Delete',
  }
  return (
    <>
     <ListPageHeader title="Example Pod List Page"/>
       <ListPageCreateDropdown createAccessReview={access}
items={items}>Actions</ListPageCreateDropdown>
     </ListPageHeader>
    </>
  );
};
```

매개변수 이름	설명
items	키:ReactNode 쌍의 드롭다운 구성 요소에 표시할 항목
onClick	드롭다운 항목을 클릭하기 위한 콜백 함수
createAccessReview	(선택 사항) 액세스 확인에 사용되는 네임스페이스 및 종류가 있는 오브젝트
children	(선택 사항) 드롭다운 토글의 하위 항목

ListPageFilter
목록 페이지에 대한 필터를 생성하는 구성 요소입니다.

예제

```
  // See implementation for more details on RowFilter and FilterValue types
  const [staticData, filteredData, onFilterChange] = useListPageFilter(
    data,
    rowFilters,
    staticFilters,
  );
  // ListPageFilter updates filter state based on user interaction and resulting filtered data can
```

```
be rendered in an independent component.
  return (
    <>
      <ListPageHeader .../>
      <ListPagBody>
        <ListPageFilter data={staticData} onFilterChange={onFilterChange} />
        <List data={filteredData} />
      </ListPageBody>
    </>
  )
```

매개변수 이름	설명
data	데이터 지점의 배열
loaded	데이터가 로드되었음을 나타냅니다.
onFilterChange	필터가 업데이트될 때의 콜백 함수
rowFilters	(선택 사항) 사용 가능한 필터 옵션을 정의하는 RowFilter 요소의 배열
nameFilterPlaceholder	(선택 사항) 이름 필터의 자리 표시자
labelFilterPlaceholder	(선택 사항) 라벨 필터의 자리 표시자
hideLabelFilter	(선택 사항) 이름 필터와 레이블 필터 대신 이름 필터만 표시합니다.
hideNameLabelFilter	(선택 사항) 이름 필터와 레이블 필터를 모두 숨깁니다.
columnLayout	(선택 사항) 열 레이아웃 오브젝트
hideColumnManagement	(선택 사항) 열 관리를 숨기는 플래그

useListPageFilter
ListPageFilter 구성 요소의 필터 상태를 관리하는 후크입니다. 모든 정적 필터, 모든 정적 및 행 필터로 필터링된 데이터, rowFilter를 업데이트하는 콜백이 포함된 튜플을 반환합니다.

예제

```
  // See implementation for more details on RowFilter and FilterValue types
  const [staticData, filteredData, onFilterChange] = useListPageFilter(
    data,
    rowFilters,
    staticFilters,
  );
  // ListPageFilter updates filter state based on user interaction and resulting filtered data can
be rendered in an independent component.
  return (
```

```
    <>
      <ListPageHeader .../>
      <ListPagBody>
        <ListPageFilter data={staticData} onFilterChange={onFilterChange} />
        <List data={filteredData} />
      </ListPageBody>
    </>
  )
```

매개변수 이름	설명
data	데이터 지점의 배열
rowFilters	(선택 사항) 사용 가능한 필터 옵션을 정의하는 RowFilter 요소의 배열
staticFilters	(선택 사항) 데이터에 정적으로 적용되는 FilterValue 요소의 배열

ResourceLink
아이콘 배지와 함께 특정 리소스 유형에 대한 링크를 생성하는 구성 요소입니다.

예제

```
  <ResourceLink
    kind="Pod"
    name="testPod"
    title={metadata.uid}
  />
```

매개변수 이름	설명
kind	(선택 사항) 리소스 종류(예: Pod, 배포, 네임스페이스)
groupVersionKind	(선택 사항) 그룹, 버전 및 종류가 있는 오브젝트
className	(선택 사항) 구성 요소에 대한 클래스 스타일
displayName	(선택 사항) 구성 요소의 표시 이름, 설정된 경우 리소스 이름을 덮어씁니다.
inline	(선택 사항) 어린이와 함께 아이콘 배지 및 이름을 인라인으로 생성하는 플래그
linkTo	(선택 사항) Link 오브젝트를 생성하는 플래그 - 기본값은 true입니다.
name	(선택 사항) 리소스 이름

매개변수 이름	설명
namesapce	(선택 사항) kind 리소스가 연결할 특정 네임스페이스
hideIcon	(선택 사항) 아이콘 배지를 숨기는 플래그
title	(선택 사항) 링크 오브젝트의 제목(표시되지 않음)
dataTest	(선택 사항) 테스트를 위한 식별자
onClick	(선택 사항) 구성 요소를 클릭할 때의 콜백 함수
truncate	(선택 사항) 너무 긴 경우 링크를 자르기 위한 플래그

ResourceIcon
특정 리소스 유형에 대한 아이콘 배지를 생성하는 구성 요소입니다.

예제

```
<ResourceIcon kind="Pod"/>
```

매개변수 이름	설명
kind	(선택 사항) 리소스 종류(예: Pod, 배포, 네임스페이스)
groupVersionKind	(선택 사항) 그룹, 버전 및 종류가 있는 오브젝트
className	(선택 사항) 구성 요소에 대한 클래스 스타일

useK8sModel
redux에서 제공된 K8sGroupVersionKind의 k8s 모델을 검색하는 후크입니다. 첫 번째 항목이 k8s 모델인 배열을 반환하고 두 번째 항목이 inFlight 상태로 반환합니다.

예제

```
const Component: React.FC = () => {
  const [model, inFlight] = useK8sModel({ group: 'app'; version: 'v1'; kind: 'Deployment' });
  return ...
}
```

매개변수 이름	설명
groupVersionKind	k8s 리소스 K8sGroupVersionKind 그룹, k8sGroupVersionKind 대신 더 이상 사용되지 않는 그룹, 버전, 즉 GVK(그룹/버전/종류) K8sResourceKindReference에 대한 참조를 전달할 수 있습니다.

useK8sModels
redux에서 모든 현재 k8s 모델을 검색하는 후크입니다. 첫 번째 항목이 k8s 모델 목록과 두 번째 항목이 있는 배열을 inFlight 상태로 반환합니다.

예제

```
const Component: React.FC = () => {
  const [models, inFlight] = UseK8sModels();
  return ...
}
```

useK8sWatchResource
로드 및 오류 상태와 함께 k8s 리소스를 검색하는 후크 첫 번째 항목이 리소스이고, 두 번째 항목이 로드된 상태이고, 세 번째 항목이 오류 상태인 배열을 반환합니다.

예제

```
const Component: React.FC = () => {
  const watchRes = {
        ...
      }
  const [data, loaded, error] = useK8sWatchResource(watchRes)
  return ...
}
```

매개변수 이름	설명
initResource	리소스를 확인하는 데 필요한 옵션.

useK8sWatchResources
로드 및 오류의 각 상태와 함께 k8s 리소스를 검색하는 후크 initResources에 제공된 대로 키가 있고 값이 로드 및 오류의 세 가지 속성 데이터를 가지는 맵을 반환합니다.

예제

```
const Component: React.FC = () => {
  const watchResources = {
        'deployment': {...},
        'pod': {...}
        ...
      }
  const {deployment, pod} = useK8sWatchResources(watchResources)
  return ...
}
```

매개변수 이름	설명
initResources	리소스는 키-값 쌍으로 조사해야 합니다. 여기서 키는 리소스에 고유하며 해당 리소스를 조사하는 데 필요한 옵션은 값입니다.

consoleFetch
콘솔 특정 헤더를 추가하고 재시도 및 타임아웃을 허용하는 fetch 에 대한 사용자 정의 래퍼를 사용하여 응답 상태 코드의 유효성을 검사하고 필요한 경우 적절한 오류를 발생시키거나 사용자를 로그아웃합니다. 응답에 해결되는 작업을 반환합니다.

매개변수 이름	설명
url	가져올 URL
options	가져오기를 위해 전달할 옵션
timeout	시간 초과(밀리초)

consoleFetchJSON
콘솔 특정 헤더를 추가하고 재시도 및 타임아웃을 허용하는 fetch 관련 사용자 정의 래퍼입니다. 또한 응답 상태 코드의 유효성을 검사하고 적절한 오류가 발생하거나 필요한 경우 사용자를 기록합니다. 응답을 JSON 오브젝트로 반환합니다. 내부적으로 consoleFetch 를 사용합니다. 응답에서 JSON 오브젝트로 확인되는 작업을 반환합니다.

매개변수 이름	설명
url	가져올 URL
method	사용할 HTTP 메서드입니다. 기본값은 GET입니다.
options	가져오기를 위해 전달할 옵션
timeout	시간 초과(밀리초)
cluster	요청할 클러스터의 이름입니다. 기본값은 사용자가 선택한 활성 클러스터입니다.

consoleFetchText
콘솔 특정 헤더를 추가하고 재시도 및 타임아웃을 허용하는 fetch 관련 사용자 정의 래퍼입니다. 또한 응답 상태 코드의 유효성을 검사하고 적절한 오류가 발생하거나 필요한 경우 사용자를 기록합니다. 응답을 텍스트로 반환합니다. 내부적으로 consoleFetch 를 사용합니다. 응답에 텍스트로 해결되는 작업을 반환합니다.

매개변수 이름	설명
url	가져올 URL
options	가져오기를 위해 전달할 옵션
timeout	시간 초과(밀리초)
cluster	요청할 클러스터의 이름입니다. 기본값은 사용자가 선택한 활성 클러스터입니다.

getConsoleRequestHeaders
현재 redux 상태를 사용하여 API 요청에 대해 가장 및 다중 클러스터 관련 헤더를 생성하는 함수입니다. redux 상태를 기준으로 적절한 가장 및 클러스터 요청 헤더가 포함된 개체를 반환합니다.

매개변수 이름	설명
targetCluster	제공된 targetCluster를 사용하여 현재 활성 클러스터 덮어쓰기

k8sGetResource
제공된 옵션에 따라 클러스터에서 리소스를 가져옵니다. 이름이 제공되면 하나의 리소스를 반환하고 다른 리소스는 모델과 일치하는 모든 리소스를 반환합니다. 이 명령은 이름이 모델에 일치하는 모든 리소스를 반환하는 경우 리소스가 포함된 JSON 오브젝트로 응답을 확인하는 것을 반환합니다. 실패하는 경우 HTTP 오류 응답을 사용하여 스키마가 거부됩니다.

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달되는 경우
options.model	k8s 모델
options.name	리소스가 제공되지 않은 경우 해당 이름과 일치하는 모든 리소스를 찾습니다.
options.ns	조사할 네임스페이스는 클러스터 범위 리소스에 지정되어서는 안 됩니다.
options.path	제공되는 경우 하위 경로로 첨부
options.queryParams	URL에 포함될 쿼리 매개변수입니다.
options.requestInit	사용할 fetch init 오브젝트입니다. 여기에는 요청 헤더, 메서드, 리디렉션 등이 있을 수 있습니다. 자세한 내용은 Interface RequestInit 에서 참조하십시오.

k8sCreateResource
제공된 옵션에 따라 클러스터에 리소스를 생성합니다. 생성된 리소스의 응답으로 확인되는 커밋을 반환합니다. 실패의 경우 HTTP 오류 응답을 사용하여 거부됩니다.

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달되는 경우
options.model	k8s 모델
options.data	생성할 리소스의 페이로드

매개변수 이름	설명
options.path	제공되는 경우 하위 경로로 첨부
options.queryParams	URL에 포함될 쿼리 매개변수입니다.

k8sUpdateResource
providedoptions를 기반으로 클러스터의 전체 리소스를 업데이트합니다. 클라이언트가 기존 리소스를 완전히 교체해야 하는 경우 k8sUpdate를 사용할 수 있습니다. 또는 k8sPatch를 사용하여 부분 업데이트를 수행할 수 있습니다. 업데이트된 리소스의 응답으로 확인되는 커밋을 반환합니다. 실패의 경우 HTTP 오류 응답을 사용하여 거부됩니다.

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달되는 경우
options.model	k8s 모델
options.data	업데이트할 k8s 리소스의 페이로드
options.ns	조사할 네임스페이스는 클러스터 범위 리소스에 지정하지 않아야 합니다.
options.name	업데이트할 리소스 이름입니다.
options.path	제공되는 경우 하위 경로로 첨부
options.queryParams	URL에 포함될 쿼리 매개변수입니다.

k8sPatchResource
제공된 옵션에 따라 클러스터의 모든 리소스를 패치합니다. 클라이언트가 부분 업데이트를 수행해야 하는 경우 k8sPatch를 사용할 수 있습니다. 또는 k8sUpdate를 사용하여 기존 리소스를 완전히 교체할 수 있습니다. 자세한 내용은 데이터 추적기 를 참조하십시오. 패치된 리소스의 응답으로 확인되는 커밋을 반환합니다. 실패의 경우 HTTP 오류 응답을 사용하여 거부됩니다.

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달됩니다.
options.model	k8s 모델
options.resource	패치할 리소스입니다.
options.data	작업, 경로 및 값을 사용하여 기존 리소스에 패치할 데이터만 적용합니다.
options.path	제공되는 경우 하위 경로로 추가합니다.

매개변수 이름	설명
options.queryParams	URL에 포함될 쿼리 매개변수입니다.

k8sDeleteResource
제공된 모델을 기반으로 클러스터에서 리소스를 삭제합니다. 가비지 컬렉션 작업은 Foreground|Background가 제공된 모델에서 propagationPolicy 속성을 사용하거나 json으로 전달할 수 있습니다. 상태 유형의 응답으로 확인되는 작업을 반환합니다. 실패의 경우 HTTP 오류 응답을 사용하여 거부됩니다.

예제

kind: 'DeleteOptions', apiVersion: 'v1', propagationPolicy

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달되는 항목입니다.
options.model	k8s 모델
options.resource	삭제할 리소스입니다.
options.path	제공되는 경우 하위 경로로 첨부
options.queryParams	URL에 포함될 쿼리 매개변수입니다.
options.requestInit	사용할 fetch init 오브젝트입니다. 여기에는 요청 헤더, 메서드, 리디렉션 등이 있을 수 있습니다. 자세한 내용은 Interface RequestInit 에서 참조하십시오.
options.json	제공되거나 모델의 "propagationPolicy"가 기본값인 경우 명시적으로 리소스의 가비지 컬렉션을 제어할 수 있습니다.

k8sListResource
제공된 옵션에 따라 리소스를 클러스터의 배열로 나열합니다. 응답에 해결되는 작업을 반환합니다.

매개변수 이름	설명
options	맵에서 키-값 쌍으로 전달되는 경우
options.model	k8s 모델
options.queryParams	쿼리 매개변수는 URL에 포함될 수 있으며 레이블 선택기의 키 "labelSelector"도 함께 전달할 수 있습니다.

매개변수 이름	설명
options.requestInit	사용할 fetch init 오브젝트입니다. 여기에는 요청 헤더, 메서드, 리디렉션 등이 있을 수 있습니다. 자세한 내용은 Interface RequestInit 에서 참조하십시오.

k8sListResourceItems
k8sListResource와 동일한 인터페이스이지만 하위 항목을 반환합니다. 모델의 apiVersion, 즉 group/version 을 반환합니다.

getAPIVersionForModel
k8s 모델에 apiVersion을 제공합니다.

매개변수 이름	설명
model	k8s 모델

getGroupVersionKindForResource
리소스에 대한 그룹, 버전 및 종류를 제공합니다. 제공된 리소스에 대한 그룹, 버전, 종류를 반환합니다. 리소스에 API 그룹이 없으면 그룹 "core"가 반환됩니다. 리소스에 잘못된 apiVersion이 있는 경우 오류가 발생합니다.

매개변수 이름	설명
resource	k8s 리소스

getGroupVersionKindForModel
k8s 모델의 그룹, 버전 및 종류를 제공합니다. 제공된 모델의 그룹, 버전, 종류를 반환합니다. 모델에 apiGroup이 없는 경우 그룹 "core"가 반환됩니다.

매개변수 이름	설명
model	k8s 모델

StatusPopupSection
팝업 창에 상태를 표시하는 구성 요소입니다. **console.dashboards/overview/health/resource** 확장을 빌드하는 데 유용한 구성 요소입니다.

예제

```
<StatusPopupSection
  firstColumn={
    <>
      <span>{title}</span>
      <span className="text-secondary">
        My Example Item
      </span>
    </>
```

```
    }
    secondColumn='Status'
  >
```

매개변수 이름	설명
firstColumn	팝업의 첫 번째 열 값
secondColumn	(선택 사항) 팝업의 두 번째 열에 대한 값
children	(선택 사항) 팝업에 대한 하위 항목

StatusPopupItem
상태 팝업에 사용되는 상태 요소; StatusPopupSection 에서 사용됩니다.

예제

```
<StatusPopupSection
  firstColumn='Example'
  secondColumn='Status'
>
  <StatusPopupItem icon={healthStateMapping[MCGMetrics.state]?.icon}>
    Complete
  </StatusPopupItem>
  <StatusPopupItem icon={healthStateMapping[RGWMetrics.state]?.icon}>
     Pending
  </StatusPopupItem>
</StatusPopupSection>
```

매개변수 이름	설명
value	(선택 사항) 표시할 텍스트 값
icon	(선택 사항) 표시할 아이콘
children	하위 요소

개요
대시보드에 대한 래퍼 구성 요소를 생성합니다.

예제

```
    <Overview>
      <OverviewGrid mainCards={mainCards} leftCards={leftCards} rightCards={rightCards} />
    </Overview>
```

매개변수 이름	설명
className	(선택 사항) div의 스타일 클래스
children	(선택 사항) 대시보드의 요소

개요 Cryostat
대시보드의 카드 요소의 Grid를 만들고 Overview 내에서 사용됩니다.

예제

```
<Overview>
  <OverviewGrid mainCards={mainCards} leftCards={leftCards} rightCards={rightCards} />
</Overview>
```

매개변수 이름	설명
mainCards	Grid 카드
leftCards	(선택 사항) Grid 왼쪽의 카드
rightCards	(선택 사항) Grid 오른쪽의 카드

InventoryItem
인벤토리 카드 항목을 생성합니다.

예제

```
return (
  <InventoryItem>
    <InventoryItemTitle>{title}</InventoryItemTitle>
    <InventoryItemBody error={loadError}>
      {loaded && <InventoryItemStatus count={workerNodes.length} icon={<MonitoringIcon />}
/>}
    </InventoryItemBody>
  </InventoryItem>
)
```

매개변수 이름	설명
children	항목 내부를 렌더링하는 요소

InventoryItemTitle
InventoryItem 내에서 사용되는 인벤토리 카드 항목에 대한 제목을 생성합니다.

예제

```
return (
```

```
  <InventoryItem>
    <InventoryItemTitle>{title}</InventoryItemTitle>
    <InventoryItemBody error={loadError}>
      {loaded && <InventoryItemStatus count={workerNodes.length} icon={<MonitoringIcon />}
/>}
    </InventoryItemBody>
  </InventoryItem>
 )
```

매개변수 이름	설명
children	제목 내에서 렌더링할 요소

InventoryItemBody
인벤토리 카드의 본문을 생성합니다. **InventoryCard** 내에서 사용되며 **InventoryTitle** 과 함께 사용할 수 있습니다.

예제

```
 return (
  <InventoryItem>
    <InventoryItemTitle>{title}</InventoryItemTitle>
    <InventoryItemBody error={loadError}>
      {loaded && <InventoryItemStatus count={workerNodes.length} icon={<MonitoringIcon />}
/>}
    </InventoryItemBody>
  </InventoryItem>
 )
```

매개변수 이름	설명
children	인벤토리 카드 또는 제목 내에서 렌더링되는 요소
error	div의 요소

InventoryItemStatus
InventoryItemBody에서 사용되는 선택적 링크 주소를 사용하여 인벤토리 카드의 개수 및 아이콘을 생성합니다.

예제

```
 return (
  <InventoryItem>
    <InventoryItemTitle>{title}</InventoryItemTitle>
    <InventoryItemBody error={loadError}>
      {loaded && <InventoryItemStatus count={workerNodes.length} icon={<MonitoringIcon />}
/>}
    </InventoryItemBody>
  </InventoryItem>
 )
```

매개변수 이름	설명
count	디스플레이 수
icon	디스플레이 아이콘
linkTo	(선택 사항) 링크 주소

InventoryItemLoading
인벤토리 카드가 로드될 때; InventoryItem 및 관련 구성 요소와 함께 사용됩니다.

예제

```
if (loadError) {
   title = <Link to={workerNodesLink}>{t('Worker Nodes')}</Link>;
} else if (!loaded) {
  title = <><InventoryItemLoading /><Link to={workerNodesLink}>{t('Worker Nodes')}</Link>
</>;
}
return (
  <InventoryItem>
    <InventoryItemTitle>{title}</InventoryItemTitle>
  </InventoryItem>
)
```

useFlag
FlexVolumeAGS redux 상태에서 지정된 기능 플래그를 반환하는 후크입니다. 요청된 기능 플래그 또는 정의되지 않은 부울 값을 반환합니다.

매개변수 이름	설명
flag	반환할 기능 플래그

CodeEditor
마우스로 도움말 및 완료가 포함된 기본 지연 로드 코드 편집기입니다.

예제

```
<React.Suspense fallback={<LoadingBox />}>
  <CodeEditor
    value={code}
    language="yaml"
  />
</React.Suspense>
```

매개변수 이름	설명
value	렌더링할 yaml 코드를 나타내는 문자열입니다.

매개변수 이름	설명
언어	편집기의 언어를 나타내는 문자열입니다.
options	Monaco 편집기 옵션입니다. 자세한 내용은 인터페이스 IStandAloneEditorConstructionOptions 를 참조하십시오.
minHeight	유효한 CSS 별점 값의 최소 편집기 높이입니다.
showShortcuts	편집기 상단에 바로 가기를 표시하는 부울입니다.
toolbarLinks	편집기 상단에 있는 툴바 링크 섹션에서 렌더링된 ReactNode 배열입니다.
Onchange	콜백은 코드 변경 이벤트입니다.
onSave	CTRL / CMD + S 명령이 트리거될 때 호출되는 콜백입니다.
ref	**{ editor?: istandaloneCodeEditor }** 에 대한 참조에 대해 반응합니다. **editor** 속성을 사용하여 편집기를 제어하는 모든 메서드에 액세스할 수 있습니다. 자세한 내용은 Interface IStandaloneCodeEditor 를 참조하십시오.

ResourceYAMLEditor
커서 도움말 및 완료를 사용하여 Kubernetes 리소스에 대한 지연 로드 YAML 편집기입니다. 구성 요소에서는 YAMLEditor를 사용하고 리소스 업데이트 처리, 경고, 저장, 취소 및 다시 로드, 액세스 가능 등과 같은 기능을 더 많이 사용합니다. onSave 콜백이 제공되지 않는 한 리소스 업데이트가 자동으로 처리됩니다. React.Suspense 구성 요소에서 래핑해야 합니다.

예제

```
<React.Suspense fallback={<LoadingBox />}>
  <ResourceYAMLEditor
    initialResource={resource}
    header="Create resource"
    onSave={(content) => updateResource(content)}
  />
</React.Suspense>
```

매개변수 이름	설명
initialResource	편집기에서 표시할 리소스를 나타내는 YAML/Object입니다. 이 prop은 초기 렌더링 중에만 사용됩니다.
header	YAML 편집기 상단에 헤더 추가

매개변수 이름	설명
onSave	저장 버튼의 콜백입니다. 전달하면 편집기에 의해 리소스에서 수행되는 기본 업데이트가 재정의됩니다.

ResourceEventStream
특정 리소스와 관련된 이벤트를 표시하는 구성 요소입니다.

예제

```
const [resource, loaded, loadError] = useK8sWatchResource(clusterResource);
return <ResourceEventStream resource={resource} />
```

매개변수 이름	설명
resource	관련 이벤트가 표시되어야 하는 오브젝트입니다.

usePrometheusPoll
단일 쿼리를 위해 Prometheus에 대한 폴링을 설정합니다. 쿼리 응답을 포함하는 튜플, 응답이 완료되었는지 여부를 나타내는 부울 플래그, 요청 또는 요청 후 처리 중에 발생한 모든 오류를 반환합니다.

매개변수 이름	설명
{PrometheusEndpoint} props.endpoint	PrometheusEndpoint 중 하나(레이블, 쿼리, 범위, 규칙, 대상)
{string} [props.query]	(선택 사항) Prometheus 쿼리 문자열 비어 있거나 정의되지 않은 경우 폴링이 시작되지 않습니다.
{number} [props.delay]	(선택 사항) 폴링 지연 간격 (ms)
{number} [props.endTime]	(선택 사항) QUERY_RANGE enpoint의 경우 쿼리 범위 끝
{number} [props.samples]	(선택 사항) QUERY_RANGE 엔드포인트
{number} [options.timespan]	(선택 사항) QUERY_RANGE 엔드포인트
{string} [options.namespace]	(선택 사항) 추가할 검색 매개 변수
{string} [options.timeout]	(선택 사항) 추가할 검색 매개 변수

Timestamp
타임스탬프를 렌더링하는 구성 요소입니다. 타임스탬프는 Timestamp 구성 요소의 제공 인스턴스 간에 동기화됩니다. 제공된 타임스탬프는 사용자 로케일에 따라 포맷됩니다.

매개변수 이름	설명
timestamp	렌더링할 타임 스탬프입니다. 형식은 ISO 8601(Kubernetes에서 사용), epoch 타임스탬프 또는 날짜 인스턴스여야 합니다.
simple	구성 요소의 간단한 버전을 생략하고 아이콘 및 툴팁을 렌더링합니다.
omitSuffix	접미사를 저장하는 날짜를 포맷합니다.
className	구성 요소에 대한 추가 클래스 이름입니다.

useModal
Modals를 시작하기 위한 후크입니다.

예제

```
const context: AppPage: React.FC = () => {<br/> const [launchModal] = useModal();<br/> const onClick = () => launchModal(ModalComponent);<br/> return (<br/>   <Button onClick= {onClick}>Launch a Modal</Button><br/> )<br/>}<br/>`
```

ActionServiceProvider
console.action/provider 확장 유형에 대한 다른 플러그인에서 기여를 받을 수 있는 구성 요소입니다.

예제

```
  const context: ActionContext = { 'a-context-id': { dataFromDynamicPlugin } };

  ...

  <ActionServiceProvider context={context}>
    {({ actions, options, loaded }) =>
      loaded && (
        <ActionMenu actions={actions} options={options} variant=
{ActionMenuVariant.DROPDOWN} />
      )
    }
  </ActionServiceProvider>
```

매개변수 이름	설명
context	contextId 및 선택적 플러그인 데이터가 있는 오브젝트

NamespaceECDHE
왼쪽 위치에 네임스페이스 드롭다운 메뉴를 사용하여 수평 툴바를 렌더링하는 구성 요소입니다. 추가 구성 요소는 자식으로 전달될 수 있으며 네임스페이스 드롭다운 오른쪽에 렌더링됩니다. 이 구성 요소는 페이지 상단에서 사용하도록 설계되었습니다. k8s 리소스가 있는 페이지의 경우와 같이 사용자가 활성 네임스페이스를 변경할 수 있어야 하는 페이지에서 사용해야 합니다.

예제

```
  const logNamespaceChange = (namespace) => console.log(`New namespace:
${namespace}`);

  ...

  <NamespaceBar onNamespaceChange={logNamespaceChange}>
    <NamespaceBarApplicationSelector />
  </NamespaceBar>
  <Page>

    ...
```

매개변수 이름	설명
onNamespaceChange	(선택 사항) 네임스페이스 옵션을 선택할 때 실행되는 함수입니다. 새 네임스페이스를 유일한 인수로 문자열 형식으로 허용합니다. 활성 네임스페이스는 옵션이 선택될 때 자동으로 업데이트되지만 이 기능을 통해 추가 논리를 적용할 수 있습니다. 네임스페이스가 변경되면 URL의 namespace 매개변수가 이전 네임스페이스에서 새로 선택한 네임스페이스로 변경됩니다.
isDisabled	(선택 사항) true로 설정된 경우 네임스페이스 드롭다운을 비활성화하는 부울 플래그입니다. 이 옵션은 네임스페이스 드롭다운에만 적용되며 하위 구성 요소에는 영향을 미치지 않습니다.
children	(선택 사항) 도구 모음 내부에서 네임스페이스 드롭다운 오른쪽에 렌더링할 추가 요소를 추가합니다.

ErrorBoundaryFallbackPage
"Oh no!를 표시하려면 전체 페이지 ErrorBoundaryFallbackPage 구성 요소를 만듭니다. 스택 추적 및 기타 유용한 디버깅 정보와 함께 메시지가 잘못되었습니다. 이는 구성 요소와 함께 Inconjunction에 사용됩니다.

예제

```
//in ErrorBoundary component
 return (
   if (this.state.hasError) {
     return <ErrorBoundaryFallbackPage errorMessage={errorString} componentStack=
{componentStackString}
       stack={stackTraceString} title={errorString}/>;
   }

   return this.props.children;
)
```

매개변수 이름	설명
errorMessage	오류 메시지에 대한 텍스트 설명
componentStack	예외의 구성 요소 추적
stack	예외 스택 추적
title	오류 경계 페이지의 헤더로 렌더링할 제목

QueryBrowser
그래프와 상호 작용하기 위한 컨트롤과 함께 Prometheus PromQL 쿼리의 결과 그래프를 렌더링하는 구성 요소입니다.

예제

```
<QueryBrowser
  defaultTimespan={15 * 60 * 1000}
  namespace={namespace}
  pollInterval={30 * 1000}
  queries={[
    'process_resident_memory_bytes{job="console"}',
    'sum(irate(container_network_receive_bytes_total[6h:5m])) by (pod)',
  ]}
/>
```

매개변수 이름	설명
customDataSource	(선택 사항) PromQL 쿼리를 처리하는 API 끝점의 기본 URL입니다. 제공되는 경우 기본 API 대신 데이터를 가져오는 데 사용됩니다.
defaultSamples	(선택 사항) 각 데이터 시리즈에 대해 표시된 기본 데이터 샘플 수입니다. 데이터 시리즈가 많은 경우 QueryBrowser는 여기에 지정된 것보다 낮은 수의 데이터 샘플을 자동으로 선택할 수 있습니다.
defaultTimespan	(선택 사항) 그래프의 기본 시간(밀리초)입니다. 기본값은 1,800,000(30분)입니다.
disabledSeries	(선택 사항) 이러한 정확한 레이블 / 값 쌍으로 데이터 시리즈를 비활성화(표시하지 않음)합니다.
disableZoom	(선택 사항) 그래프 확대 컨트롤을 비활성화하는 플래그입니다.
filterLabels	(선택 사항) 필요한 경우 반환된 데이터 시리즈를 이러한 라벨 / 값 쌍과 일치하는 것으로 필터링합니다.

매개변수 이름	설명
fixedEndTime	(선택 사항) 데이터를 현재 시간으로 표시하는 대신 표시된 시간 범위의 종료 시간을 설정합니다.
formatSeriesTitle	(선택 사항) 단일 데이터 시리즈의 제목으로 사용할 문자열을 반환합니다.
GraphLink	(선택 사항) 다른 페이지에 대한 링크를 렌더링하는 구성 요소(예: 이 쿼리에 대한 자세한 정보 가져오기)
hideControls	(선택 사항) 그래프 시간 간격을 변경하기 위한 그래프 컨트롤을 숨기는 플래그 등입니다.
isStack	(선택 사항) 선 그래프 대신 누적 그래프를 표시하는 플래그입니다. showStackedControl이 설정된 경우에도 사용자가 줄 그래프로 전환할 수 있습니다.
네임스페이스	(선택 사항) 제공되는 경우 이 네임스페이스에 대해서만 데이터가 반환됩니다(이 네임스페이스 레이블이 있는 시리즈만).
OnZoom	(선택 사항) 그래프를 확대할 때 호출되는 콜백입니다.
pollInterval	(선택 사항) 설정하는 경우 최신 데이터(밀리초)를 표시하도록 그래프가 업데이트되는 빈도를 결정합니다.
queries	그래프에 결과를 실행하고 표시하는 PromQL 쿼리의 배열입니다.
showLegend	(선택 사항) 그래프 아래에 범례를 표시할 수 있는 플래그입니다.
showStackedControl	누적 그래프 모드와 선 그래프 모드 간 전환을 위한 그래프 컨트롤을 표시할 수 있는 플래그입니다.
Cryostat	(선택 사항) 그래프에서 밀리초 단위로 처리해야 하는 시간 간격입니다.
units	(선택 사항) Y축 및 툴팁에 표시할 단위입니다.

useAnnotationsModal
Kubernetes 리소스 주석 편집을 위한 모달을 시작하는 콜백을 제공하는 후크입니다.

예제

```
const PodAnnotationsButton = ({ pod }) => {
  const { t } = useTranslation();
  const launchAnnotationsModal = useAnnotationsModal<PodKind>(pod);
```

```
  return <button onClick={launchAnnotationsModal}>{t('Edit Pod Annotations')}</button>
}
```

매개변수 이름	설명
resource	K8sResourceCommon 유형의 오브젝트에 대한 주석을 편집할 리소스입니다.

반환

리소스의 주석을 편집하기 위한 모달을 시작하는 함수입니다.

useDeleteModal
리소스 삭제를 위한 모달을 시작하는 콜백을 제공하는 후크입니다.

예제

```
const DeletePodButton = ({ pod }) => {
  const { t } = useTranslation();
  const launchDeleteModal = useDeleteModal<PodKind>(pod);
  return <button onClick={launchDeleteModal}>{t('Delete Pod')}</button>
}
```

매개변수 이름	설명
resource	삭제할 리소스입니다.
redirectTo	(선택 사항) 리소스를 삭제한 후 리디렉션할 위치입니다.
message	(선택 사항) 모달에 표시할 메시지입니다.
btnText	(선택 사항) 삭제 버튼에 표시할 텍스트입니다.
deleteAllResources	(선택 사항) 동일한 종류의 모든 리소스를 삭제하는 함수입니다.

반환

리소스를 삭제하기 위한 모달을 시작하는 함수입니다.

useLabelsModel
Kubernetes 리소스 레이블 편집을 위한 모달을 시작하는 콜백을 제공하는 후크입니다.

예제

```
const PodLabelsButton = ({ pod }) => {
  const { t } = useTranslation();
  const launchLabelsModal = useLabelsModal<PodKind>(pod);
```

```
  return <button onClick={launchLabelsModal}>{t('Edit Pod Labels')}</button>
}
```

매개변수 이름	설명
resource	K8sResourceCommon 유형의 오브젝트인 레이블을 편집할 리소스입니다.

반환

리소스의 레이블을 편집하기 위한 모달을 시작하는 함수입니다.

useActiveNamespace
활성 네임스페이스를 설정하기 위한 현재 활성 네임스페이스 및 콜백을 제공하는 후크입니다.

예제

```
const Component: React.FC = (props) => {
  const [activeNamespace, setActiveNamespace] = useActiveNamespace();
  return <select
    value={activeNamespace}
    onChange={(e) => setActiveNamespace(e.target.value)}
  >
    {
      // ...namespace options
    }
  </select>
}
```

반환

현재 활성 네임스페이스 및 설정자 콜백을 포함하는 튜플입니다.

useUserSettings
사용자 설정 값을 설정하기 위한 사용자 설정 값과 콜백을 제공하는 후크입니다.

예제

```
const Component: React.FC = (props) => {
  const [state, setState, loaded] = useUserSettings(
    'devconsole.addPage.showDetails',
    true,
    true,
  );
  return loaded ? (
     <WrappedComponent {...props} userSettingState={state} setUserSettingState={setState} />
   ) : null;
};
```

반환

사용자 설정 vauel, setter 콜백 및 로드된 부울이 포함된 튜플입니다.

useQuickStartContext

현재 퀵 스타트 컨텍스트 값을 제공하는 후크입니다. 이를 통해 플러그인은 콘솔 빠른 시작 기능과 상호 작용할 수 있습니다.

예제

```
const OpenQuickStartButton = ({ quickStartId }) => {
  const { setActiveQuickStart } = useQuickStartContext();
  const onClick = React.useCallback(() => {
    setActiveQuickStart(quickStartId);
  }, [quickStartId]);
  return <button onClick={onClick}>{t('Open Quick Start')}</button>
};
```

Returns

빠른 시작 컨텍스트 값 오브젝트입니다.

PerspectiveContext
더 이상 사용되지 않음: 제공된 **usePerspectiveContext** 를 대신 사용합니다. 화면 컨텍스트를 생성합니다.

매개변수 이름	설명
PerspectiveContextType	활성 화면 및 설정자가 있는 오브젝트

useAccessReviewAllowed
더 이상 사용되지 않음: 대신 **@console/dynamic-plugin-sdk** 에서 **useAccessReview** 를 사용합니다. 지정된 리소스에 대한 사용자 액세스 권한을 허용하는 후크입니다. **isAllowed** 부울 값을 반환합니다.

매개변수 이름	설명
resourceAttributes	액세스 검토를 위한 리소스 속성
impersonate	명의 도용 세부 정보

useSafetyFirst
더 이상 사용되지 않음: 이 후크는 콘솔 기능과 관련이 없습니다. 지정된 구성 요소를 마운트 해제할 수 있는 경우 React 상태의 안전한 비동기식 설정을 보장하는 후크입니다. 상태 값과 집합 함수의 쌍을 포함하는 배열을 반환합니다.

매개변수 이름	설명
initialState	초기 상태 값

YAMLEditor
더 이상 사용되지 않음: 마우스 가리키기 및 완료가 포함된 기본 지연 로드된 YAML 편집기입니다.

예제

```
<React.Suspense fallback={<LoadingBox />}>
  <YAMLEditor
```

```
    value={code}
  />
</React.Suspense>
```

매개변수 이름	설명
value	렌더링할 yaml 코드를 나타내는 문자열입니다.
options	Monaco 편집기 옵션입니다.
minHeight	유효한 CSS 별점 값의 최소 편집기 높이입니다.
showShortcuts	편집기 상단에 바로 가기를 표시하는 부울입니다.
toolbarLinks	편집기 상단에 있는 툴바 링크 섹션에서 렌더링된 ReactNode 배열입니다.
Onchange	콜백은 코드 변경 이벤트입니다.
onSave	CTRL / CMD + S 명령이 트리거될 때 호출되는 콜백입니다.
ref	**{ editor?: istandaloneCodeEditor }** 에 대한 참조에 대해 반응합니다. **editor** 속성을 사용하여 편집기를 제어하는 모든 메서드에 액세스할 수 있습니다.

4.5.3. 동적 플러그인 문제 해결

플러그인을 로드하는 데 문제가 발생하면 이 문제 해결 팁 목록을 참조하십시오.

- **다음 명령을 실행하여 콘솔 Operator 구성에서 플러그인을 활성화했으며 플러그인 이름이 출력인지 확인합니다.**

  ```
  $ oc get console.operator.openshift.io cluster -o jsonpath='{.spec.plugins}'
  ```

  - **관리자 화면의 개요 페이지의 상태 카드에서 활성화된 플러그인을 확인합니다. 플러그인이 최근에 활성화된 경우 브라우저를 새로 고쳐야 합니다.**

- **다음을 통해 플러그인 서비스가 정상인지 확인합니다.**
  - **플러그인 포드 상태가 실행 중이며 컨테이너가 준비되었는지 확인합니다.**
  - **서비스 레이블 선택기가 Pod와 일치하고 대상 포트가 올바른지 확인합니다.**
  - **콘솔 Pod의 터미널에 있는 서비스의 plugin-manifest.json 을 컬거나 클러스터의 다른 Pod를 컬링합니다.**

- **ConsolePlugin 리소스 이름(consolePlugin.name)이 package.json 에 사용된 플러그인 이름과 일치하는지 확인합니다.**

- **ConsolePlugin 리소스에서 서비스 이름, 네임스페이스, 포트 및 경로가 올바르게 선언되었는지 확인합니다.**

- 플러그인 서비스에서 HTTPS 및 서비스 제공 인증서를 사용하는지 확인합니다.
- 콘솔 pod 로그에서 인증서 또는 연결 오류를 확인합니다.
- 플러그인이 사용하는 기능 플래그가 비활성화되었는지 확인합니다.
- 플러그인이 **package.json** 에 충족되지 않은**consolePlugin.dependencies** 가 없는지 확인합니다.
  - 여기에는 콘솔 버전 종속 항목 또는 다른 플러그인의 종속성이 포함될 수 있습니다. 브라우저에서 플러그인의 이름으로 JS 콘솔을 필터링하여 기록된 메시지를 확인합니다.
- nav 확장 화면 또는 섹션 ID에 오타가 없는지 확인합니다.
  - 플러그인이 로드될 수 있지만 ID가 올바르지 않으면 nav 항목이 누락되었습니다. URL을 편집하여 플러그인 페이지로 직접 이동해 보십시오.
- 콘솔 포드에서 플러그인 서비스로 트래픽을 차단하는 네트워크 정책이 없는지 확인합니다.
  - 필요한 경우 openshift-console 네임스페이스의 콘솔 Pod가 서비스에 요청할 수 있도록 네트워크 정책을 조정합니다.
- 개발자 도구 브라우저의 콘솔 탭에서 브라우저에 로드할 동적 플러그인 목록을 확인합니다.
  - **window.SERVER_FLAGS.consolePlugins** 를 평가하여 콘솔 프런트 엔드의 동적 플러그인을 확인합니다.

5장. 웹 터미널

5.1. 웹 터미널 설치

AWS OperatorHub의 Red Hat OpenShift Service에 나열된 Web Terminal Operator를 사용하여 웹 터미널을 설치할 수 있습니다. Web Terminal Operator를 설치하면 **DevWorkspace** CRD와 같은 명령행 구성에 필요한 사용자 정의 리소스 정의(CRD)가 자동으로 설치됩니다. 웹 콘솔은 웹 터미널을 열 때 필요한 리소스를 생성합니다.

사전 요구 사항

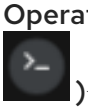
- AWS 웹 콘솔에서 Red Hat OpenShift Service에 로그인되어 있습니다.
- 클러스터 관리자 권한이 있어야 합니다.

프로세스

1. 웹 콘솔의 관리자 화면에서 Operator → OperatorHub로 이동합니다.
2. 키워드로 필터링 상자를 사용하여 카탈로그에서 Web Terminal Operator를 검색한 다음Web Terminal 타일을 클릭합니다.
3. Web Terminal 페이지에서 Operator에 대한 간략한 설명을 확인하고Install을 클릭합니다.
4. Install Operator 페이지에서 모든 필드의 기본값을 유지합니다.
   - Update Channel 메뉴의 fast 옵션으로 Web Terminal Operator의 최신 릴리스버전을 설치할 수 있습니다.
   - Installation Mode 메뉴의 All namespaces on the cluster를 사용하면 Operator가 클러스터의 모든 네임스페이스를 모니터링하고 사용할 수 있습니다.
   - Installed Namespace 메뉴의 openshift-operators 옵션은 기본 **openshift-operators** 네임스페이스에 Operator를 설치합니다.
   - Approval Strategy 메뉴의 Automatic 옵션은 Operator에 향후 지원되는 업그레이드가 OLM(Operator Lifecycle Manager)에 의해 자동으로 처리됩니다.
5. 설치를 클릭합니다.
6. Installed Operators 페이지에서 View Operator를 클릭하여 Operator가 Installed Operators 페이지에 나열되어 있는지 확인합니다.

참고

Web Terminal Operator는 DevWorkspace Operator를 종속성으로 설치합니다.

7. Operator가 설치되면 페이지를 새로 고침하여 콘솔 마스트 헤드에 있는 명령행 터미널 아이콘()을 확인합니다.

5.2. 웹 터미널 사용

웹 콘솔에서 포함된 명령행 터미널 인스턴스를 시작할 수 있습니다. 이 터미널 인스턴스는 **oc,kubectl,odo,kn,tkn,helm** 및 **subctl** 과 같은 클러스터와 상호 작용하기 위한 일반적인 CLI 툴로 사전 설

치됩니다. 또한 작업 중인 프로젝트의 컨텍스트도 포함되어 있으며 인증 정보를 사용하여 자동으로 로그인됩니다.

5.2.1. 웹 터미널 액세스

Web Terminal Operator가 설치되면 웹 터미널에 액세스할 수 있습니다. 웹 터미널이 초기화되면 **oc,kubectl,odo,kn,tkn,helm** 및 **subctl** 과 같은 사전 설치된 CLI 툴을 웹 터미널에서 사용할 수 있습니다. 터미널에서 실행한 명령 목록에서 명령을 선택하여 명령을 다시 실행할 수 있습니다. 이러한 명령은 여러 터미널 세션에 걸쳐 유지됩니다. 웹 터미널은 종료되거나 브라우저 창 또는 탭을 닫을 때까지 열린 상태로 유지됩니다.

사전 요구 사항

- AWS 클러스터의 Red Hat OpenShift Service에 액세스할 수 있으며 웹 콘솔에 로그인되어 있습니다.
- Web Terminal Operator가 클러스터에 설치되어 있습니다.

프로세스

1. 웹 터미널을 시작하려면 콘솔 마스트 헤드에서 명령행 터미널 아이콘()을 클릭합니다. 명령줄 터미널 창에 웹 터미널 인스턴스가 표시됩니다. 이 인스턴스는 사용자의 인증 정보를 사용하여 자동으로 로그인됩니다.
2. 현재 세션에서 프로젝트를 선택하지 않은 경우 프로젝트 드롭다운 목록에서 **DevWorkspace** CR을 생성해야 하는 프로젝트를 선택합니다. 기본적으로 현재 프로젝트는 선택됩니다.

참고

- 하나의 **DevWorkspace** CR은 한 사용자의 웹 터미널을 정의합니다. 이 CR에는 사용자의 웹 터미널 상태 및 컨테이너 이미지 구성 요소에 대한 세부 정보가 포함되어 있습니다.
- **DevWorkspace** CR은 아직 존재하지 않는 경우에만 생성됩니다.

3. Start를 클릭하여 선택한 프로젝트를 사용하여 웹 터미널을 초기화합니다.
4. + 를 클릭하여 콘솔의 웹 터미널에서 여러 탭을 엽니다.

5.3. 웹 터미널 문제 해결

5.3.1. 웹 터미널 및 네트워크 정책

클러스터에 네트워크 정책이 구성된 경우 웹 터미널이 시작되지 않을 수 있습니다. 웹 터미널 인스턴스를 시작하려면 Web Terminal Operator가 웹 터미널의 pod와 통신하여 실행 중인지 확인해야 하며 AWS 웹 콘솔의 Red Hat OpenShift Service는 터미널 내의 클러스터에 자동으로 로그인할 수 있는 정보를 보내야 합니다. 두 단계 모두 실패하면 웹 터미널이 시작되지 않고 컨텍스트 데드라인 초과 오류가 발생할 때까지 터미널 패널이 로드 상태에 있습니다.

이 문제를 방지하려면 터미널에 사용되는 네임스페이스에 대한 네트워크 정책에서 **openshift-console** 및 **openshift-operators** 네임스페이스에서 수신을 허용하는지 확인합니다.

다음 샘플에서는 **openshift-console** 및 **openshift-operators** 네임스페이스에서 수신할 수 있는 **NetworkPolicy** 오브젝트를 보여줍니다.

openshift-console 네임스페이스에서 수신 허용

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-openshift-console
spec:
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: openshift-console
  podSelector: {}
  policyTypes:
  - Ingress
```

openshift-operators 네임스페이스에서 수신 허용

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-openshift-operators
spec:
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: openshift-operators
  podSelector: {}
  policyTypes:
  - Ingress
```

5.4. 웹 터미널 설치 제거

Web Terminal Operator를 설치 제거해도 Operator를 설치할 때 생성된 CRD(사용자 정의 리소스 정의) 또는 관리 리소스는 제거되지 않습니다. 보안을 위해 이러한 구성 요소를 수동으로 제거해야 합니다. 이러한 구성 요소를 제거하면 Operator가 제거될 때 터미널이 유휴 상태가 되지 않기 때문에 클러스터 리소스를 저장합니다.

웹 터미널 설치 제거는 2단계로 수행됩니다.

1. Operator를 설치할 때 추가된 Web Terminal Operator 및 관련 사용자 지정 리소스(CR)를 제거합니다.
2. Web Terminal Operator의 종속성으로 추가된 DevWorkspace Operator 및 관련 사용자 정의 리소스를 설치 제거합니다.

5.4.1. Web Terminal Operator 제거

Web Terminal Operator 및 Operator에서 사용하는 사용자 정의 리소스를 제거하여 웹 터미널을 설치 제거할 수 있습니다.

사전 요구 사항

- 클러스터 관리자 권한이 있는 AWS 클러스터의 Red Hat OpenShift Service에 액세스할 수 있습니다.
- **oc** CLI를 설치했습니다.

프로세스

1. 웹 콘솔의 관리자 화면에서 Operator → Installed Operators로 이동합니다.
2. 필터 목록을 스크롤하거나 이름으로 필터링 상자에 키워드를 입력하여 Web Terminal Operator를 찾습니다.
3. Web Terminal Operator의 옵션 메뉴  를 클릭한 다음 Operator 설치 제거를 선택합니다.
4. Uninstall Operator 확인 대화 상자에서 Uninstall을 클릭하여 클러스터에서 Operator, Operator 배포 및 pod를 제거합니다. Operator는 실행을 중지하고 더 이상 업데이트가 수신되지 않습니다.

5.4.2. DevWorkspace Operator 제거

웹 터미널을 완전히 제거하려면 Operator에서 사용하는 DevWorkspace Operator 및 사용자 정의 리소스도 제거해야 합니다.

중요

DevWorkspace Operator는 독립 실행형 Operator이며 클러스터에 설치된 다른 Operator의 종속성으로 필요할 수 있습니다. DevWorkspace Operator가 더 이상 필요하지 않은 경우에만 아래 단계를 수행하십시오.

사전 요구 사항

- 클러스터 관리자 권한이 있는 AWS 클러스터의 Red Hat OpenShift Service에 액세스할 수 있습니다.
- **oc** CLI를 설치했습니다.

프로세스

1. 관련 Kubernetes 오브젝트와 함께 Operator에서 사용하는 **DevWorkspace** 사용자 정의 리소스를 제거합니다.

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

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

주의

이 단계가 완료되지 않으면 종료자가 Operator를 완전히 제거하기 어렵습니다.

2. 나머지 서비스, 시크릿 및 구성 맵을 제거합니다. 설치에 따라 다음 명령에 포함된 일부 리소스가 클러스터에 존재하지 않을 수 있습니다.

```
$ 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 clusterrole devworkspace-webhook-server
```

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

3. DevWorkspace Operator를 설치 제거합니다.

   a. 웹 콘솔의 관리자 화면에서 Operator → Installed Operators로 이동합니다.

   b. 필터 목록을 스크롤하거나 이름으로 필터링 상자에서 키워드를 입력하여 DevWorkspace Operator를 찾습니다.

   c. Operator의 옵션 메뉴  를 클릭한 다음 Operator 설치 제거를 선택합니다.

   d. Operator 설치 제거 확인 대화 상자에서 설치 제거를 클릭하여 클러스터에서 Operator, Operator 배포 및 pod를 제거합니다. Operator는 실행을 중지하고 더 이상 업데이트가 수신되지 않습니다.

6장. 퀵 스타트 튜토리얼 정보

AWS 웹 콘솔에서 Red Hat OpenShift Service에 대한 퀵 스타트 튜토리얼을 생성하는 경우 다음 지침에 따라 모든 퀵 스타트에서 사용자 환경을 일관되게 유지합니다.

6.1. 퀵 스타트 이해

퀵 스타트는 사용자 작업에 대한 가이드 튜토리얼입니다. 웹 콘솔의 Help 메뉴에서 퀵 스타트에 액세스할 수 있습니다. 애플리케이션, Operator 또는 기타 다른 제품 오퍼링을 사용하는 경우에 유용합니다.

퀵 스타트는 주로 작업과 단계로 구성됩니다. 각 작업에는 여러 단계가 있으며 각 퀵스타트에는 여러 작업이 있습니다. 예를 들면 다음과 같습니다.

- 작업 1
  - 1 단계
  - 2 단계
  - 3 단계
- 작업 2
  - 1 단계
  - 2 단계
  - 3 단계
- 작업 3
  - 1 단계
  - 2 단계
  - 3 단계

6.2. 사용자 워크플로우 퀵 스타트

기존 퀵 스타트 튜토리얼과 상호 작용할 때 예상되는 워크플로우 환경은 다음과 같습니다.

1. 관리자 또는 개발자 화면에서 도움말 아이콘을 클릭하고 퀵 스타트를 선택합니다.
2. 퀵 스타트 카드를 클릭합니다.
3. 표시되는 창에서 Start를 클릭합니다.
4. 화면에 있는 지침을 작성한 후 다음을 클릭합니다.
5. 표시되는 Check your work 모듈에서 질문에 대답하여 작업을 완료했는지 확인합니다.
   a. Yes를 선택한 경우 Next를 클릭하여 다음 작업으로 이동합니다.
   b. No를 선택하면 작업 단계를 반복하고 작업을 다시 확인하십시오.
6. 위의 1단계에서 6단계를 반복하여 퀵 스타트의 나머지 작업을 완료합니다.

7. 마지막 작업이 완료되면 Close를 클릭하여 퀵 스타트를 종료합니다.

6.3. 퀵 스타트 구성 요소

퀵 스타트는 다음 섹션으로 구성됩니다.

- Card: 제목, 설명, 시간 커밋 및 완료 상태를 포함하여 퀵 스타트의 기본 정보를 제공하는 카탈로그 타일
- Introduction: 퀵 스타트의 목표 및 작업에 대한 간략한 개요
- Task headings: 빠른 시작의 각 작업에 대한 하이퍼 링크 제목
- Check your work module: 사용자가 작업을 완료했는지 확인할 수 있는 모듈입니다. 퀵 스타트의 다음 작업으로 이동하기 전에 작업이 성공적으로 완료되었는지 확인할 수 있습니다.
- Hint: 사용자가 제품의 특정 영역을 식별하는 데 도움이 되는 애니메이션
- Button
  - Next and back buttons: 퀵 스타트의 각 작업 내에서 단계 및 모듈로 이동하기 위한 버튼
  - Final screen buttons: 퀵 스타트를 위한 버튼, 퀵 스타트의 이전 작업으로 돌아가거나 퀵 스타트를 표시할 수 있는 버튼

퀵 스타트의 주요 콘텐츠 영역에는 다음 섹션이 포함되어 있습니다.

- Card copy
- Introduction
- Task steps
- Modals and in-app messaging
- 작업 모듈 확인