OpenShift Container Platform 4.9

웹 콘솔

OpenShift Container Platform에서 웹 콘솔 시작하기

Last Updated: 2023-04-08

OpenShift Container Platform 4.9 웹 콘솔

OpenShift Container Platform에서 웹 콘솔 시작하기

법적 공지

초록

이 문서는 OpenShift Container Platform 웹 콘솔 액세스 및 사용자 지정에 대한 정보를 제공합니다.

차례

1장. 웹 콘솔 개요 **3**
1.1. 웹 콘솔의 관리자 화면 정보 3
1.2. 웹 콘솔의 DEVELOPER 모드 정보 3
1.3. 관점 액세스 4

2장. 웹 콘솔에 액세스 **6**
2.1. 전제 조건 6
2.2. 웹 콘솔 이해 및 액세스 6

3장. OPENSHIFT CONTAINER PLATFORM 대시 보드를 사용하여 클러스터 정보 검색 **7**
3.1. OPENSHIFT CONTAINER PLATFORM 대시 보드 페이지 정보 7

4장. 사용자 기본 설정 추가 **8**
4.1. 사용자 기본 설정 설정 8

5장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 구성 **9**
5.1. 전제 조건 9
5.2. 웹 콘솔 구성 9

6장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 사용자 정의 **10**
6.1. 사용자 정의 로고 및 제품 이름 추가 10
6.2. 웹 콘솔에서 사용자 정의 링크 작성 11
6.3. 콘솔 경로 사용자 정의 12
6.4. 로그인 페이지 사용자 정의 14
6.5. 외부 로그 링크의 템플릿 정의 15
6.6. 사용자 정의 알림 배너 만들기 16
6.7. CLI 다운로드 사용자 정의 16
6.8. KUBERNETES 리소스에 YAML 예제 추가 17

7장. 웹 콘솔의 웹 터미널 정보 **19**
7.1. 웹 터미널 설치 19
7.2. 웹 터미널 사용 20
7.3. 웹 터미널 설치 제거 20

8장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 비활성화 **24**
8.1. 사전 요구 사항 24
8.2. 웹 콘솔 비활성화 24

9장. 웹 콘솔에서 퀵 스타트 튜토리얼 만들기 **25**
9.1. 퀵 스타트 이해 25
9.2. 사용자 워크플로우 퀵 스타트 25
9.3. 퀵 스타트 구성 요소 26
9.4. 퀵 스타트 사용 26
9.5. 퀵 스타트 콘텐츠 가이드 라인 38
9.6. 추가 리소스 41

1장. 웹 콘솔 개요

Red Hat OpenShift Container Platform 웹 콘솔은 그래픽 사용자 인터페이스를 제공하여 프로젝트 데이터를 시각화하고 관리, 관리 및 문제 해결 작업을 수행할 수 있습니다. 웹 콘솔은 openshift-console 프로젝트의 컨트롤 플레인 노드에서 Pod로 실행됩니다. **console-operator** Pod에서 관리합니다. **Administrator** 및 **Developer** 모드 둘 다 지원됩니다.

Administrator 및 **Developer** 모드 둘 다 OpenShift Container Platform에 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 사용 정보, 중요한 이벤트 스트림을 볼 수 있으므로 계획 및 문제 해결 작업을 단순화할 수 있습니다. 프로젝트 관리자와 클러스터 관리자는 모두 **관리자** 화면을 볼 수 있습니다.

클러스터 관리자는 OpenShift Container Platform 4.7 이상에서 웹 터미널 Operator를 사용하여 포함된 명령줄 터미널 인스턴스를 열 수도 있습니다.

참고

표시되는 기본 웹 콘솔 모드 정보는 사용자의 역할에 따라 다르게 표시됩니다. 사용자가 관리자로 인식되면 기본적으로 **Administrator** 관점이 표시됩니다.

관리자 관점은 다음과 같은 관리자의 유스 케이스에 특정한 워크 플로우를 제공합니다.

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

추가 리소스

웹 터미널 Operator에 대한 자세한 내용은 웹 콘솔 의 웹 터미널 정보를 참조하십시오.

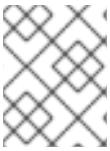
1.2. 웹 콘솔의 DEVELOPER 모드 정보

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

- 구성 요소에서 롤링의 실시간 시각화를 보고 롤아웃을 다시 생성합니다.

- 애플리케이션 상태, 리소스 사용률, 프로젝트 이벤트 스트리밍, 할당량 소비를 확인합니다.
- 프로젝트를 다른 사람과 공유합니다.
- 프로젝트에 대한 Prometheus Query Language(PromQL) 쿼리를 실행하고 플롯에서 시각화된 지표를 검사하여 애플리케이션의 문제를 해결합니다. 지표는 클러스터 상태 및 모니터링 중인 모든 사용자 정의 워크로드에 대한 정보를 제공합니다.

클러스터 관리자는 OpenShift Container Platform 4.7 이상의 웹 콘솔에서 포함된 명령줄 터미널 인스턴스를 열 수도 있습니다.



참고

표시되는 기본 웹 콘솔 모드 정보는 사용자의 역할에 따라 다르게 표시됩니다. 사용자가 Developer로 인식되면 기본적으로 **Developer** 모드 정보가 표시됩니다.

Developer 모드는 다음과 같은 개발자의 유스 케이스에 특정한 워크 플로우를 제공합니다.

- 기존 코드베이스, 이미지 및 컨테이너 파일을 가져와 OpenShift Container Platform에서 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. 관점 액세스

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

전제 조건

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

추가 리소스

관점 변경에 대한 자세한 내용은 사용자 환경 추가를 참조하십시오.

프로세스

1. 관점 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 관리자에 대해 자세히 알아보기
- **개발자** 화면을 사용하여 OpenShift Container Platform에서 애플리케이션 생성 및 배포
- **토폴로지 보기에서** 프로젝트에 애플리케이션을 표시, 배포 상태 확인 및 애플리케이션과 상호 작용
- 클러스터 정보 보기
- 웹 콘솔 구성
- 웹 콘솔 사용자 정의
- 웹 콘솔에서 포함된 명령줄 터미널 인스턴스 시작
- 퀵 스타트 튜토리얼 만들기
- 웹 콘솔 비활성화

2장. 웹 콘솔에 액세스

OpenShift Container Platform 웹 콘솔은 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 마스터에서 Pod로 실행됩니다. Pod에서는 웹 콘솔을 실행하는 데 필요한 정적 환경을 제공합니다. **openshift-install create cluster** 를 사용하여 OpenShift Container Platform을 성공적으로 설치한 후 설치 프로그램의 CLI 출력에서 웹 콘솔의 URL 및 설치된 클러스터의 로그인 인증 정보를 찾습니다. 예를 들면 다음과 같습니다.

출력 예

```
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>
```

이러한 세부 사항을 사용하여 로그인하고 웹 콘솔에 로그인하고 액세스하십시오.

설치하지 않은 기존 클러스터의 경우 **oc whoami --show-console**을 사용하여 웹 콘솔 URL을 확인할 수 있습니다.

3장. OPENSHIFT CONTAINER PLATFORM 대시 보드를 사용하여 클러스터 정보 검색

OpenShift Container Platform 대시 보드에는 클러스터에 대한 고급 정보가 포함되어 있으며 OpenShift Container Platform 웹 콘솔에서 **Home → Dashboards → Overview**를 통해 액세스합니다.

OpenShift Container Platform 대시 보드는 별도의 대시 보드 카드에 표시되는 다양한 클러스터 정보를 제공합니다.

3.1. OPENSHIFT CONTAINER PLATFORM 대시 보드 페이지 정보

OpenShift Container Platform 대시 보드는 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 나열되는 클러스터의 베어 메탈 호스트 (**metal3** 환경에서만 사용 가능)
- **Cluster Capacity** 차트는 관리자가 클러스터에 추가 리소스가 필요한 시기를 파악하는 데 도움이 됩니다. 이 차트에는 내부 링과 외부링이 포함되어 있으며 내부 링은 현재 소비를 표시하는 외부 링은 다음 정보를 포함하여 리소스에 설정된 임계 값을 표시합니다.
  - CPU 시간
  - 메모리 할당
  - 소비된 스토리지
  - 소비된 네트워크 리소스
- **Cluster Utilization**은 관리자가 리소스의 높은 소비 규모 및 빈도를 이해하는데 도움이 되도록 지정된 기간 동안 다양한 리소스의 용량을 표시합니다.
- **Events**는 Pod 생성 또는 다른 호스트로의 가상 머신 마이그레이션과 같은 클러스터의 최근 활동과 관련된 메시지를 표시합니다.
- **Top Consumers** 관리자가 클러스터 리소스가 소비되는 방식을 이해하는 데 도움이 됩니다. 리소스를 클릭하면 지정된 클러스터 리소스 (CPU, 메모리 또는 스토리지)를 가장 많이 소비하는 pod 및 노드가 나열된 자세한 정보가 있는 페이지로 이동합니다.

4장. 사용자 기본 설정 추가

요구 사항에 맞게 프로필의 기본 기본 설정을 변경할 수 있습니다. 기본 프로젝트, 토폴로지 보기(그래프/리스트), 미디어 편집(form 또는 YAML) 및 언어 기본 설정을 설정할 수 있습니다.

사용자 기본 설정에 대한 변경 사항은 자동으로 저장됩니다.

4.1. 사용자 기본 설정 설정

클러스터의 기본 사용자 기본 설정을 설정할 수 있습니다.

프로세스

1. 로그인 인증 정보를 사용하여 OpenShift Container Platform 웹 콘솔에 로그인합니다.
2. masthead를 사용하여 사용자 프로필 아래의 사용자 기본 설정에 액세스합니다.
3. **일반** 섹션에서 다음을 수행합니다.
   a. **Perspective** 필드에서 로그인할 기본 모드를 설정할 수 있습니다. 필요에 따라 **관리자** 또는 **개발자** 화면을 선택할 수 있습니다. 관점을 선택하지 않으면 마지막으로 방문한 화면에 기록됩니다.
   b. **프로젝트** 필드에서 작업할 프로젝트를 선택합니다. 로그인할 때마다 콘솔은 프로젝트로 기본 설정됩니다.
   c. **토폴로지 필드에서 토폴로지** 보기를 기본적으로 그래프 또는 목록 보기로 설정할 수 있습니다. 선택하지 않은 경우 콘솔은 기본적으로 마지막으로 사용한 보기로 설정됩니다.
   d. **Create/Edit resource method** 필드에서 리소스를 생성하거나 편집하는 기본 설정을 설정할 수 있습니다. 양식 및 YAML 옵션을 모두 사용할 수 있는 경우 콘솔은 기본적으로 선택 사항으로 설정됩니다.
4. 언어 섹션에서 **기본 브라우저 언어를** 선택하여 기본 브라우저 언어 설정을 사용합니다. 그렇지 않으면 콘솔에 사용할 언어를 선택합니다.

5장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 구성

OpenShift Container Platform 웹 콘솔을 변경하여 로그 아웃 리디렉션 URL을 설정하거나 콘솔을 비활성화할 수 있습니다.

5.1. 전제 조건

- OpenShift Container Platform 클러스터를 배포합니다.

5.2. 웹 콘솔 구성

console.config.openshift.io 리소스를 편집하여 웹 콘솔을 설정할 수 있습니다.

- **console.config.openshift.io** 리소스를 편집합니다.

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

다음 예제는 콘솔의 리소스 정의입니다.

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

(1) 사용자가 웹 콘솔에서 로그 아웃할 때 로드할 페이지의 URL을 지정합니다. 값을 지정하지 않으면 사용자는 웹 콘솔의 로그인 페이지로 돌아갑니다. **logoutRedirect** URL을 지정하면 사용자가 아이덴티티 공급자를 통해 단일 로그 아웃 (SLO)을 수행하여 단일 사인온 세션을 삭제할 수 있습니다.

(2) 웹 콘솔 URL입니다. 사용자 정의 값으로 업데이트하려면 **웹 콘솔 URL 사용자 정의**를 참조하십시오.

6장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 사용자 정의

OpenShift Container Platform 웹 콘솔을 사용자 정의하여 사용자 정의 로고, 제품 이름, 링크, 알림 및 명령 줄 다운로드를 설정할 수 있습니다. 이는 웹 콘솔을 특정 기업 또는 정부의 요구 사항에 맞게 조정해야 하는 경우 특히 유용합니다.

6.1. 사용자 정의 로고 및 제품 이름 추가

사용자 정의 로고 또는 사용자 정의 제품 이름을 추가하여 사용자 정의 브랜딩을 만들 수 있습니다. 이 설정은 서로 독립적이므로 모두 또는 하나씩 따로 설정할 수 있습니다.

전제 조건

- 클러스터 관리자 권한이 있어야합니다.
- 사용할 로고 파일을 만듭니다. 로고는 GIF, JPG, PNG 또는 SVG를 포함한 일반적인 이미지 형식의 파일 일 수 있으며 **max-height 60px**로 제한됩니다.

프로세스

1. **openshift-config** 네임 스페이스의 로고 파일을 설정 맵으로 가져옵니다.

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

작은 정보

다음 YAML을 적용하여 구성 맵을 만들 수 있습니다.

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

1 유효한 base64로 인코딩된 로고를 제공합니다.

2. **customLogoFile** 및 **customProductName**을 포함하도록 웹 콘솔의 Operator 설정을 편집합니다.

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

```
apiVersion: operator.openshift.io/v1
kind: Console
metadata:
  name: cluster
spec:
  customization:
    customLogoFile:
```

```
      key: console-custom-logo.png
      name: console-custom-logo
    customProductName: My Console
```

Operator 설정이 업데이트되면 사용자 정의 로고 설정 맵을 콘솔 네임 스페이스에 동기화하고 이를 콘솔 pod에 마운트한 후 다시 배포합니다.

3. 성공적으로 실행되었는지 확인합니다. 문제가 있는 경우 콘솔 클러스터 Operator는 **Degraded** 상태를 보고하고 콘솔 Operator 설정도 **CustomLogoDegraded** 상태를 **KeyOrFilenameInvalid** 또는 **NoImageProvided**와 같은 이유와 함께 보고합니다.
clusteroperator를 확인하려면 다음을 실행합니다.

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

콘솔 Operator 설정을 확인하려면 다음을 실행합니다.

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

6.2. 웹 콘솔에서 사용자 정의 링크 작성

사전 요구 사항

- 클러스터 관리자 권한이 있어야합니다.

절차

1. **Administration → Custom Resource Definitions**에서 **ConsoleLink**를 클릭합니다.
2. **Instances** 탭을 선택합니다.
3. **Create Console Link**를 클릭하고 파일을 편집합니다.

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

1 유효한 위치 설정은 **HelpMenu**, **UserMenu**, **ApplicationMenu** 및 **NamespaceDashboard**입니다.

모든 네임 스페이스에 사용자 정의 링크를 표시하려면 다음 예제를 따르십시오.

```
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
```

일부 네임 스페이스에만 사용자 정의 링크를 표시하려면 다음 예제를 따르십시오.

```
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
```

애플리케이션 메뉴에 사용자 정의 링크를 표시하려면 다음 예제를 따르십시오.

```
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. **저장을** 클릭하여 변경 사항을 적용합니다.

6.3. 콘솔 경로 사용자 정의

console 및 **downloads**의 경우 사용자 지정 경로 기능은 **ingress** 구성 경로 구성 API를 사용합니다. **console** 사용자 정의 경로가 **ingress** 구성 및 **console-operator** 구성에 모두 설정된 경우 새 **ingress** 구성 사용자 정의 경로 구성이 우선합니다. **console-operator** 구성을 통한 경로 구성은 더 이상 사용되지 않습니다.

6.3.1. 콘솔 경로 사용자 정의

클러스터 **Ingress** 구성의 **spec.componentRoutes** 필드에 사용자 정의 호스트 이름과 TLS 인증서를 설정하여 콘솔 경로를 사용자 지정할 수 있습니다.

사전 요구 사항

- 관리 권한이 있는 사용자로 클러스터에 로그인했습니다.
- TLS 인증서 및 키가 포함된 **openshift-config** 네임스페이스에 시크릿을 생성했습니다. 사용자 지정 호스트 이름 접미사가 클러스터 도메인 접미사와 일치하지 않는 경우 이 작업이 필요합니다. 접미사가 일치하는 경우 시크릿은 선택 사항입니다.

작은 정보

oc create secret tls 명령을 사용하여 TLS 시크릿을 생성할 수 있습니다.

절차

1. 클러스터 **Ingress** 구성을 편집합니다.

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

2. 사용자 정의 호스트 이름과 서비스 인증서 및 키를 설정합니다.

```
apiVersion: config.openshift.io/v1
kind: Ingress
metadata:
  name: cluster
spec:
  componentRoutes:
    - name: console
      namespace: openshift-console
      hostname: <custom_hostname> 1
      servingCertKeyPairSecret:
        name: <secret_name> 2
```

1. 사용자 지정 호스트 이름.
2. TLS 인증서 (**tls.crt**) 및 키 (**tls.key**)가 포함된 **openshift-config** 네임스페이스의 시크릿에 대한 참조입니다. 사용자 지정 호스트 이름 접미사가 클러스터 도메인 접미사와 일치하지 않는 경우 이 작업이 필요합니다. 접미사가 일치하는 경우 시크릿은 선택 사항입니다.

3. 파일을 저장하여 변경 사항을 적용합니다.

6.3.2. 다운로드 경로 사용자 지정

클러스터 **Ingress** 구성의 **spec.componentRoutes** 필드에 사용자 정의 호스트 이름과 TLS 인증서를 설정하여 다운로드 경로를 사용자 지정할 수 있습니다.

사전 요구 사항

- 관리 권한이 있는 사용자로 클러스터에 로그인했습니다.
- TLS 인증서 및 키가 포함된 **openshift-config** 네임스페이스에 시크릿을 생성했습니다. 사용자 지정 호스트 이름 접미사가 클러스터 도메인 접미사와 일치하지 않는 경우 이 작업이 필요합니다. 접미사가 일치하는 경우 시크릿은 선택 사항입니다.

작은 정보

oc create secret tls 명령을 사용하여 TLS 시크릿을 생성할 수 있습니다.

프로세스

1. 클러스터 **Ingress** 구성을 편집합니다.

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

2. 사용자 정의 호스트 이름과 서비스 인증서 및 키를 설정합니다.

```
apiVersion: config.openshift.io/v1
kind: Ingress
metadata:
  name: cluster
spec:
  componentRoutes:
    - name: downloads
      namespace: openshift-console
      hostname: <custom_hostname> 1
      servingCertKeyPairSecret:
        name: <secret_name> 2
```

1. 사용자 지정 호스트 이름.
2. TLS 인증서 (**tls.crt**) 및 키 (**tls.key**)가 포함된 **openshift-config** 네임스페이스의 시크릿에 대한 참조입니다. 사용자 지정 호스트 이름 접미사가 클러스터 도메인 접미사와 일치하지 않는 경우 이 작업이 필요합니다. 접미사가 일치하는 경우 시크릿은 선택 사항입니다.

3. 파일을 저장하여 변경 사항을 적용합니다.

6.4. 로그인 페이지 사용자 정의

사용자 정의 로그인 페이지를 사용하여 서비스 약관 정보를 작성하십시오. 사용자 정의 로그인 페이지는 GitHub 또는 Google과 같은 타사 로그인 공급자를 사용하는 경우에도 사용자가 신뢰하는 브랜드 페이지를 표시하고 사용자를 인증 공급자로 리디렉션하는데 유용할 수 있습니다. 인증 프로세스 중에 사용자 정의 오류 페이지를 렌더링할 수도 있습니다.

참고

오류 템플릿 사용자 지정은 요청 헤더 및 OIDC 기반 IDP와 같이 리디렉션을 사용하는 ID 프로바이더(IDP)로 제한됩니다. LDAP 및 htpasswd와 같이 직접 암호 인증을 사용하는 IDP에는 영향을 미치지 않습니다.

전제 조건

- 클러스터 관리자 권한이 있어야합니다.

프로세스

1. 다음 명령을 실행하여 수정할 수 있는 템플릿을 만듭니다.

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

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

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

2. 시크릿을 사용하여 해당 정보를 만듭니다:

```
$ 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. 다음을 실행합니다.

```
$ oc edit oauths cluster
```

4. 사양을 업데이트합니다.

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

oc explain oauths.spec.templates을 실행하여 옵션을 파악합니다.

6.5. 외부 로그 링크의 템플릿 정의

로그를 찾는 데 도움이되는 서비스에 연결되어 있지만 특정 방식으로 URL을 생성해야 하는 경우 링크의 템플릿을 정의할 수 있습니다.

전제 조건

- 클러스터 관리자 권한이 있어야합니다.

프로세스

1. **Administration → Custom Resource Definitions**에서 **ConsoleExternalLogLink**를 클릭합니다.
2. **Instances** 탭을 선택합니다.
3. **Create Console External Log Link**를 클릭하고 파일을 편집합니다.

```
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
```

6.6. 사용자 정의 알림 배너 만들기

전제 조건

- 클러스터 관리자 권한이 있어야합니다.

프로세스

1. **Administration → Custom Resource Definitions**에서 **ConsoleNotification**을 클릭합니다.
2. **Instances** 탭을 선택합니다.
3. **Create Console Notification**을 클릭하고 파일을 편집합니다.

```
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) 유효한 위치 설정은 **BannerTop**, **BannerBottom** 및 **BannerTopBottom** 입니다.

4. **만들기를** 클릭하여 변경 사항을 적용합니다.

6.7. CLI 다운로드 사용자 정의

파일 패키지 또는 패키지를 제공하는 외부 페이지를 직접 지정할 수있는 사용자 정의 링크 텍스트 및 URL을 사용하여 CLI를 다운로드하기 위한 링크를 설정할 수 있습니다.

전제 조건

- 클러스터 관리자 권한이 있어야합니다.

프로세스

1. **Administration → Custom Resource Definitions**로 이동합니다.
2. CRD (Custom Resource Definitions) 목록에서 **ConsoleCLIDownload**를 선택합니다.

3. **YAML** 탭을 클릭한 후 편집합니다.

```
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** 버튼을 클릭합니다.

6.8. KUBERNETES 리소스에 YAML 예제 추가

언제든지 Kubernetes 리소스에 YAML 예제를 동적으로 추가할 수 있습니다.

전제 조건

- 클러스터 관리자 권한이 있어야합니다.

프로세스

1. **Administration → Custom Resource Definitions**에서 **ConsoleYAMLSample**을 클릭합니다.
2. **YAML**을 클릭하고 파일을 편집합니다.

```
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.snippet을 사용하여 YAML 샘플이 전체 YAML 리소스 정의가 아니라 사용자 커서에서 기존 YAML 문서에 삽입할 수 있는 조각을 보여줍니다.

3. **Save**를 클릭합니다.

7장. 웹 콘솔의 웹 터미널 정보

웹 콘솔에서 포함된 명령줄 터미널 인스턴스를 시작할 수 있습니다. 웹 터미널을 사용하려면 Web Terminal Operator를 설치해야 합니다.

참고

클러스터 관리자는 OpenShift Container Platform 4.7 이상에서 웹 터미널에 액세스할 수 있습니다.

이 터미널 인스턴스는 **oc**,**kubectl**,**odo**,**kn**,**tkn**,**helm**,**kubens**,**subctl**, **kubectx** 와 같은 클러스터와 상호 작용하기 위한 일반적인 CLI 툴로 사전 설치되어 있습니다. 또한 작업 중인 프로젝트의 컨텍스트도 포함되어 있으며 인증 정보를 사용하여 자동으로 로그인됩니다.

중요

웹 터미널은 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

7.1. 웹 터미널 설치

OpenShift Container Platform OperatorHub에 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

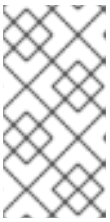
- **cluster-admin** 권한이 있는 계정을 사용하여 OpenShift Container Platform 클러스터에 액세스할 수 있습니다.

절차

1. 웹 콘솔의 **Administrator** 모드에서 **Operator → OperatorHub**로 이동합니다.
2. **Filter by keyword** 상자를 사용하여 카탈로그에서 **Web Terminal** Operator를 검색한 다음 **Web Terminal** 타일을 클릭합니다.
3. **Web Terminal** 페이지에서 Operator에 대한 간략한 설명을 확인하고 **Install**을 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 cluster**를 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. **설치**를 클릭합니다.
6. **Installed Operators** 페이지에서 **View Operator**를 클릭하여 Operator가 **Installed Operators** 페이지에 나열되어 있는지 확인합니다.



참고

OpenShift Container Platform 4.8 이전에는 Web Terminal Operator가 모든 기능을 단일 Operator에 번들했습니다. OpenShift Container Platform 4.8부터 Web Terminal Operator는 동일한 기능을 제공하기 위해 DevWorkspace Operator를 종속 항목으로 설치합니다.

7. Operator가 설치되면 페이지를 새로고침하여 콘솔 오른쪽에 있는 명령행 터미널 아이콘을 확인합니다.

7.2. 웹 터미널 사용

Web Terminal Operator가 설치되면 다음과 같이 웹 터미널을 사용할 수 있습니다.

1. 웹 터미널을 시작하려면 콘솔 오른쪽 상단에 있는 명령행 터미널 아이콘()을 클릭합니다. **명령줄 터미널 창**에 웹 터미널 인스턴스가 표시됩니다. 이 인스턴스는 사용자의 인증 정보를 사용하여 자동으로 로그인됩니다.
2. 프로젝트 드롭다운 목록에서 **DevWorkspace** CR을 생성해야 하는 **프로젝트를** 선택합니다. 기본적으로 현재 프로젝트는 선택됩니다.

참고

- **DevWorkspace** CR은 아직 존재하지 않는 경우에만 생성됩니다.
- **openshift-terminal** 프로젝트는 클러스터 관리자에게 사용되는 기본 프로젝트입니다. 다른 프로젝트를 선택할 수 있는 옵션이 없습니다.

3. **Start**를 클릭하여 선택한 프로젝트를 사용하여 웹 터미널을 초기화합니다.

웹 터미널이 초기화된 후 **oc**,**kubectl**,**odo**,**kn**,**tkn**,**helm**,**kubens**,**subctl**, **kubectx** 와 같은 사전 설치된 CLI 툴을 사용할 수 있습니다.

7.3. 웹 터미널 설치 제거

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.

Web Terminal Operator를 설치 제거해도 Operator가 설치될 때 생성된 CRD(사용자 정의 리소스 정의) 또는 관리 리소스 정의는 제거되지 않습니다. 이러한 구성 요소는 보안 목적으로 수동으로 설치 제거해야 합니다. 이러한 구성 요소를 제거하면 Operator가 제거될 때 터미널이 유휴 상태가 되지 않도록하여 클러스터 리소스를 절약할 수 있습니다.

전제 조건

- **cluster-admin** 권한이 있는 계정을 사용하여 OpenShift Container Platform 클러스터에 액세스할 수 있습니다.

7.3.1. Web Terminal Operator 및 이를 지원하는 사용자 정의 리소스 제거

콘솔과 CLI를 사용하여 Web Terminal Operator를 설치하는 동안 생성된 기존 웹 터미널 및 CR을 삭제합니다.

참고

OpenShift Container Platform 4.8 이전에는 Web Terminal Operator에서 다른 CRD를 사용하여 웹 터미널 기능을 제공합니다. Web Terminal Operator 버전 1.2.1 이하를 제거하려면 OpenShift Container Platform 4.7 설명서를 참조하십시오.

절차

1. 웹 콘솔을 사용하여 Web Terminal Operator를 설치 제거합니다.
   a. 웹 콘솔의 **Administrator** 모드에서 **Operator → Installed Operators**로 이동합니다.
   b. 필터 목록을 스크롤하거나 **Filter by name** 상자에서 키워드를 입력하여 **Web Terminal** Operator를 찾습니다.
   c. Web Terminal Operator의 옵션 메뉴  를 클릭한 다음 **Uninstall Operator** 를 선택합니다.
   d. **Uninstall Operator** 확인 대화 상자에서 **Uninstall**을 클릭하여 클러스터에서 Operator, Operator 배포 및 pod를 제거합니다. Operator는 실행을 중지하고 더 이상 업데이트가 수신되지 않습니다.
2. Operator에서 사용하는 CR을 제거합니다.

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

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

7.3.2. DevWorkspace Operator 종속성 삭제

CLI를 사용하여 Web Terminal Operator를 설치하는 동안 생성된 CRD(사용자 정의 리소스 정의) 및 추가 리소스를 삭제합니다.

중요

DevWorkspace Operator는 독립 실행형 Operator로 작동하며 클러스터에 설치된 다른 Operator의 종속성으로 필요할 수 있습니다(예: CodeReady Workspaces Operator가 종속될 수 있음). DevWorkspace Operator가 더 이상 필요하지 않은 경우에만 아래 단계를 수행합니다.

절차

1. 배포와 같은 관련 Kubernetes 오브젝트와 함께 Operator에서 사용하는 **DevWorkspace** 사용자 지정 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 제거할 수 없습니다.

2. Operator에서 사용하는 CRD를 제거합니다.

주의

DevWorkspace Operator는 변환 Webhook를 사용하는 CRD(사용자 정의 리소스 정의)를 제공합니다. 이러한 CRD를 제거하지 않으면 클러스터에 문제가 발생할 수 있습니다.

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

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

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

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

3. 관련된 모든 사용자 정의 리소스 정의가 제거되었는지 확인합니다. 다음 명령은 결과를 표시하지 않아야 합니다.

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

4. **devworkspace-webhook-server** 배포, 변경 및 검증 웹 후크를 제거합니다.

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

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

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

참고

변경 및 검증 Webhook를 제거하지 않고 **devworkspace-webhook-server** 배포를 제거하면 **oc exec** 명령을 사용하여 클러스터의 컨테이너에서 명령을 실행할 수 없습니다. 웹 후크를 제거한 후에는 **oc exec** 명령을 다시 사용할 수 있습니다.

5. 나머지 서비스, 보안 및 구성 맵을 제거합니다. 설치에 따라 다음 명령에 포함된 일부 리소스가 클러스터에 없을 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 configmap devworkspace-controller -n openshift-operators
```

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

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

6. 웹 콘솔을 사용하여 Operator 설치 제거

   a. 웹 콘솔의 **Administrator** 모드에서 **Operator → Installed Operators**로 이동합니다.

   b. 필터 목록을 스크롤하거나 **Filter by name** 상자에서 키워드를 입력하여 **DevWorkspace** Operator를 찾습니다.

   c. DevWorkspace Operator의 옵션 메뉴  를 클릭한 다음 **Uninstall Operator** 를 선택합니다.

   d. **Uninstall Operator** 확인 대화 상자에서 **Uninstall**을 클릭하여 클러스터에서 Operator, Operator 배포 및 pod를 제거합니다. Operator는 실행을 중지하고 더 이상 업데이트가 수신되지 않습니다.

8장. OPENSHIFT CONTAINER PLATFORM에서 웹 콘솔 비활성화

OpenShift Container Platform 웹 콘솔을 비활성화할 수 있습니다.

8.1. 사전 요구 사항

- OpenShift Container Platform 클러스터를 배포합니다.

8.2. 웹 콘솔 비활성화

console.operator.openshift.io 리소스를 편집하여 웹 콘솔을 비활성화할 수 있습니다.

- **console.operator.openshift.io** 리소스를 편집합니다.

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

다음 예제는 리소스에서 수정할 수 있는 매개 변수를 표시합니다.

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

1 웹 콘솔을 비활성화하려면 **managementState** 매개 변수 값을 **Removed**로 설정합니다. 이 매개 변수의 다른 유효한 값은 **Managed** (클러스터 제어 하에서 콘솔을 활성화) 및 **Unmanaged** (사용자가 웹 콘솔 관리를 제어하고 있음)입니다.

9장. 웹 콘솔에서 퀵 스타트 튜토리얼 만들기

OpenShift Container Platform 웹 콘솔에 대한 퀵 스타트 튜토리얼을 생성하려면 다음 지침에 따라 모든 퀵 스타트에서 사용자에게 일관된 사용자 환경을 유지합니다.

9.1. 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

9.2. 사용자 워크플로우 퀵 스타트

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

1. **Administrator** 또는 **Developer** 모드에서 **Help 아이콘을** 클릭하고 **Quick Starts**를 선택합니다.
2. 퀵 스타트 카드를 클릭합니다.
3. 표시되는 창에서 **Start**를 클릭합니다.
4. 화면에 있는 지침을 실행 완료한 후 **Next**를 클릭합니다.
5. 표시되는 **Check your work** 모듈에서 질문에 대답하여 작업을 완료했는지 확인합니다.
   a. **Yes**를 선택한 경우 **Next**를 클릭하여 다음 작업으로 이동합니다.
   b. **No**를 선택하면 작업 단계를 반복하고 작업을 다시 확인하십시오.
6. 위의 1단계에서 6단계를 반복하여 퀵 스타트의 나머지 작업을 완료합니다.

7. 마지막 작업이 완료되면 **Close**를 클릭하여 퀵 스타트를 종료합니다.

9.3. 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**
- **Check your work module**

9.4. 퀵 스타트 사용

OpenShift Container Platform에는 **ConsoleQuickStart** 개체에 정의된 퀵 스타트 사용자 정의 리소스가 있습니다. Operator 및 관리자는 이 리소스를 사용하여 클러스터에 퀵 스타트를 제공할 수 있습니다.

사전 요구 사항

- 클러스터 관리자 권한이 있어야합니다.

절차

1. 새로운 퀵 스타트를 만들려면 다음을 실행합니다.

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

2. 다음을 실행합니다.

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

3. 이 문서에 설명된 지침을 사용하여 YAML 파일을 업데이트합니다.
4. 편집한 내용을 저장합니다.

9.4.1. 퀵 스타트 API 문서 보기

절차

- 퀵 스타트 API 문서를 보려면 다음을 실행합니다.

```
$ oc explain consolequickstarts
```

oc explain -h를 실행하여 **oc explain** 사용법에 대한 자세한 내용을 확인합니다.

9.4.2. 퀵 스타트의 요소에서 퀵 스타트 CR에 매핑

이 섹션에서는 웹 콘솔 내에서 퀵 스타트 부분에 표시되는 퀵 스타트 리소스(CR)의 일부를 시각적으로 매핑하는 방법을 설명합니다.

9.4.2.1. conclusion 요소

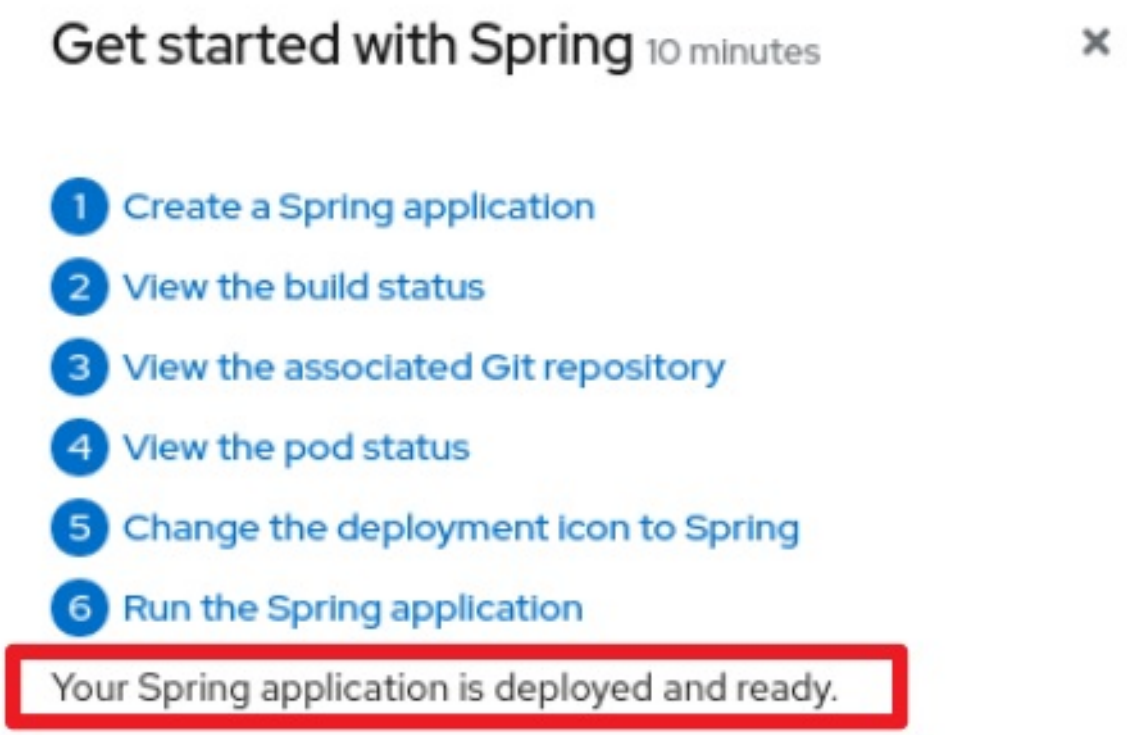
YAML 파일에 conclusion 요소 표시

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

(1) conclusion 텍스트

웹 콘솔에서 conclusion 요소의 표시

퀵 스타트의 마지막 부분에 conclusion가 표시됩니다.



9.4.2.2. description 요소

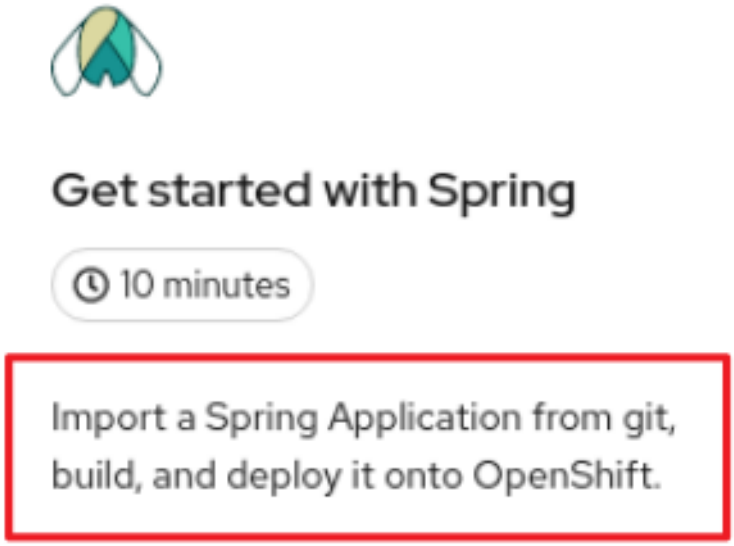
YAML 파일에서 description 요소 보기

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

1 description 텍스트

웹 콘솔에서 description 요소 보기

설명은 **퀵 스타트** 페이지의 퀵 스타트 소개 부분에 표시됩니다.



9.4.2.3. displayName 요소

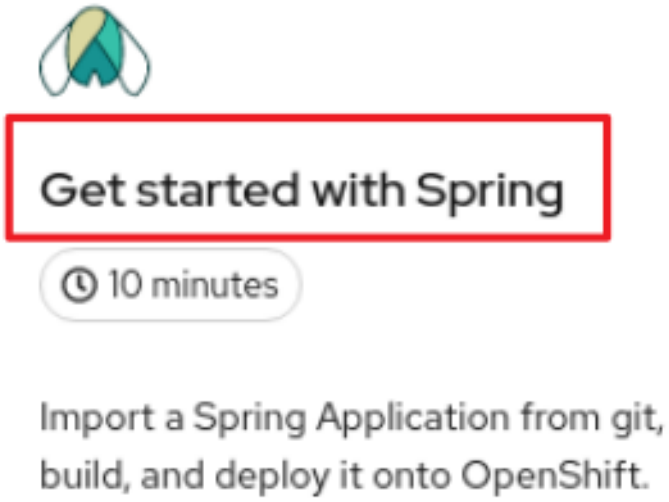
YAML 파일에서 displayName 요소 보기

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

(1) **displayName** 텍스트입니다.

웹 콘솔에서 displayName 요소 표시

표시 이름은 **퀵 스타트** 페이지의 퀵 스타트 소개 부분에 표시됩니다.



9.4.2.4. durationMinutes 요소

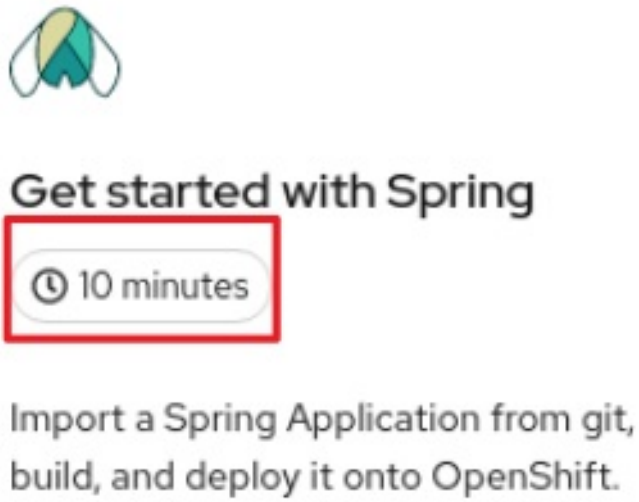
YAML 파일에서 durationMinutes element 요소 표시

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

(1) **durationMinutes** 값 (분)입니다. 이 값은 퀵 스타트를 완료하는 데 걸리는 시간을 정의합니다.

웹 콘솔에서 durationMinutes 요소 표시

duration minutes 요소는 **Quick Starts** 페이지의 퀵 스타트 소개 부분에 표시됩니다.



9.4.2.5. icon 요소

YAML 파일의 icon 요소 표시

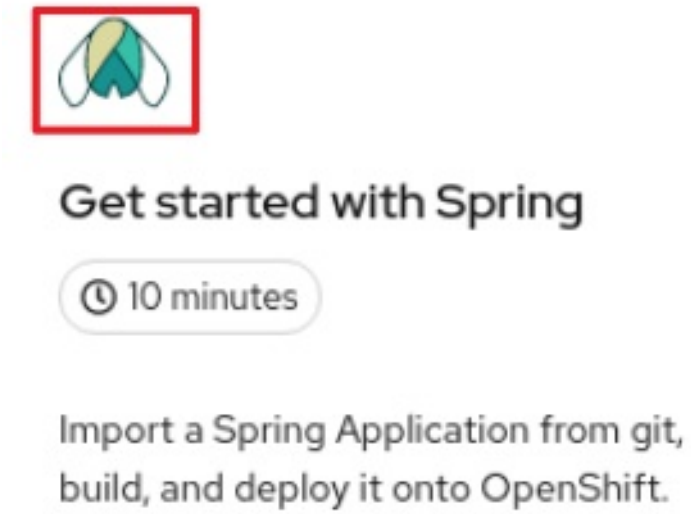
```
...
spec:
  description: 'Import a Spring Application from git, build, and deploy it onto OpenShift.'
  displayName: Get started with Spring
  durationMinutes: 10
  icon: >-   (1)


JMYXllcl8xIiBkYXRhLW5hbWU9IkxheWVyIDEiIHZpZXdCb3g9IjAgMCAxMDI0IDEwMjQiPjxkZWZzPjxzd
HlsZT4uY2xzLTF7ZmlsbDojMTUzZDNjO30uY2xzLTJ7ZmlsbDojZDhkYTlkO30uY2xzLTN7ZmlsbDojNT
hjMGE4O30uY2xzLTR7ZmlsbDojZmZmO30uY2xzLTV7ZmlsbDojM2Q5MTkxO308L3N0eWxlPjwvZGV
mcz48dGl0bGU+c25vd2Ryb3BfaWNvbl9yZ2JfZGVmYXVsdDwvdGl0bGU+PHBhdGggY2xhc3M9ImNsc
y0xIiBkPSJNMTAxMi42OSw1OTNjLTExLjEyLTM4LjA3LTMxLTczLTU5LjIxLTEwMy44LTkuNS0xMS4zL
TIzLjIxLTI4LjI5LTM5LjA2LTQ3Ljk0QzgzMy41MywzNDEsNzQ1LjM3LDIzNC4xOCw2NzQsMTY4Ljk0Yy
01LTUuMjYtMTAuMjYtMTAuMzEtMTUuNjUtMTUuMDdhMjQ2LjQ5LDI0Ni40OSwwLDAsMC0zNi41NS
0yNi44LDE4Mi41LDE4Mi41LDAsMCwwLTIwLjMtMTEuNzcsMjAxLjUzLDIwMS41MywwLDAsMC00My4
xOS0xNUExNTUuMjQsMTU1LjI0LDAsMCwwLDUyOCw5NS4yYy02Ljc2LS42OC0xMS43NC0uODEtM
TQuMzktLjgxaDBsLTEuNjIsMC0xLjYyLDBhMTc3LjMsMTc3LjMsMCwwLDAtMzEuNzcsMy4zNSwyMDg
uMjMsMjA4LjIzLDAsMCwwLTU2LjEyLDE3LjU2LDE4MSwxODEsMCwwLDAtMjAuMjcsMTEuNzUsMjQ
3LjQzLDI0Ny40MywwLDAsMC0zNi41NywyNi44MUMzNjAuMjUsMTU4LjYyLDM1NSwxNjMuNjgsMzUw
LDE2OWMtNzEuMzUsNjUuMjUtMTU5LjUsMTcyLTI0MC4zOSwyNzIuMjhDOTMuNzMsNDYwLjg4LDg
wLDQ3Ny44Nyw3MC41Miw0ODkuMTcsNDIuMzUsNTIwLDIyLjQzLDU1NC45LDExLjMxLDU5MywuNz
IsNjI5LjIyLTEuNzMsNjY3LjY5LDQsNzA3LjMxLDE1LDc4Mi40OSw1NS43OCw4NTkuMTIsMTE4LjkzLD
kyMy4wOWEyMiwyMiwwLDAsMCwxNS41OSw2LjUyaDEuODNsMS44Ny0uMzJjODEuMDYtMTMuOT
EsMTEwLTc5LjU3LDE0My40OC0xNTUuNiwzLjkxLTguODgsNy45NS0xOC4wNSwxMi4yLTI3LjQzcTU
uNDIsOC41NCwxMS4zOSwxNi4yM2MzMS44NSw0MC45MSw3NS4xMiw2NC42NywxMzIuMzIsNzIuNj
NsMTguOCwyLjYyLDQuOTUtMTguMzNjNjMuMjYtNDkuMDcsMzUuMy05MC44NSw1MC42NC0xMT
YuMTksMTUuMzQsMjUuMzQsMzcuMzgsNjcuMTIsNTAuNjQsMTE2LjE5bDUsMTguMzMsMTguOC0yL
jYyYzU3LjItOCwxMDAuNDctMzEuNzIsMTMyLjMyLTcyLjYzLDQuMTMtNi4yMzM0MjI2LjQ4LTI1LjI3LjE1
DkuMzgsOC4yOSwxOC41NSwxMi4yLDI3LjQzLDMzLjQ5LDc2LDYyLjQyLDE0MS42OSwxNDMuNDgs
MTU1LjZsMS44MS4zMWgxLjg5YTIyLDIyLDAsMCwwLDE1LjU5LTYuNTJjNjMuMTUtNjQsMTAzLjk1LT
E0MC42LDExNC44OS0yMTUuNzhDMTAyNS43Myw2NjcuNjksMTAyMy4yOCw2MjkuMjIsMTAxMi42O
Sw1OTNaIi8+PHBhdGggY2xhc3M9ImNscy0yIiBkPSJNMzY0LjE1LDE4NS4yM2MxNy44OS0xNi40LDM
0LjctMzAuMTUsNDkuNzctNDAuMTFhMjEyLDIxMiwwLDAsMSw2NS45My0yNS43M0ExOTgsMTk4LDA
sMCwxLDUxMiwxMTYuMjdhMTk2LjExLDE5Ni4xMSwwLDAsMSwzMiwzLjFjNC41LjkxLDkuMzYsMi4wN
wxNC41MywzLjUyLDYwLjQxLDIwLjQ4LDg0LjkyLDkxLjA1LTQ3LjQ0LDI0OC4wNi0yOC43NSwzNC4x
Mi0xNDAuNywxOTQuODQtMTg0LjY2LDI2OC40MzAuODYsNjMwLjg2LDAsMCwwLTMzLjIyLDU4Ljg3LD
U4LjMyQzI3Niw2NTUuMzQsMjY1LjQsNTk4LDI2NS40LDUyMC4yOSwyNjUuNCwzNDAuNjEsMzExLjY
5LDI0MC43NCwzNjQuMTUsMTg1LjIzWiIvPjxwYXRoIGNsYXNzPSJjbHMtMyIgZD0iTTUyNy41NCwzO
DQuODNjODQuMDYtOTkuNywxMTYuMDYtMTc3LjI4LDk1LjIyLTIzMC43NCwxMS42Miw4LjY5LDI0LD
E5LjIsMzcuMDYsMzEuMTMsNTIuNDgsNTUuNSw5OC43OCwxNTUuMzgsOTguNzgsMzM1LjA3LDAAs
NzcuNzEtMTAuNiwxMzUuMDUtMjcuNzcsMTc3LjI4LjczLDYyLjk4OC43MywwLDAsMC0zMy4yMy01OC
4zMmMtMzkuNjUtNjUuMjYtMTMxLjQ1LTE5OS0xNzEuOTMtMjUyLjI3QzUyNi4zMywzODYuMjksNTI3LDM4
NS41Miw1MjcuNTQsMzg0LjgzWiIvPjxwYXRoIGNsYXNzPSJjbHMtNCIgZD0iTTEzNC41OCw5MDguM
DdoLS4wNmEuMzkuMzksMCwwLDEtLjI3LS4xMWMtMTE5LjUyLTEyMS4wNy0xNTUtMjg3LjQtNDcuN
TQtNDA0LjU4LDM0LjYzLTQxLjE0LDEyMC0xNTEuNiwyMDIuNzUtMjQyLjE5LTMuMTMsMTMuNy02LjQ
0LjI1LTguOTIsMjEuNjktMjQuMzQsNjQuNDUtMzYuMzYsNjcuNjcsMTQ0LjMyLTM2LjY3LDIzNy40MSwwLDU2Lj
zLDUuNTgsMTA2LDE2LjU5LDE0Ny4xNEEzMDcuNDksMzA3LjQ5LDAsMCwwLDI4MC45MSw3MjND
MjM3LDgxNi44OCwyMTYuOTMsODk0LjA5LDEzNC41OCw5MDguMDdaIi8+PHBhdGggY2xhc3M9ImN
scy01IiBkPSJNNTgzLjQzLDgxMy43OUM1NjAuMTgsNzI3LjcyLDUxMiw2NjQuNC4xNMtNDguMTcsNjMuNTctNzEuNDMsMTQ5LjY0Yy00OC40NS02LjQ0LTEwMC45MS0yNy41Mi0xMzUu
NjYtOTEuMThhNjQ1LjY4LDY0NS42OCwwLDAsMSwzOS41Ny03MS41NGwuMjMtLjMyLjE5LjE5MS4zM2M2M
```

```
zOC02My42MywxMjYuNC0xOTEuMzcsMTY3LjEyLTI0NS42Niw0MC43MSw1NC4yOCwxMjkuMSwxO
DIsMTY3LjEyLDI0NS42NmwuMTkuMzMuMjEuMzJhNjQ1LjY4LDY0NS42OCwwLDAsMSwzOS41Nyw
3MS41NEM2ODQuMzQsNzg2LjI3LDYzMS44OCw4MDcuMDUsNTgzLjQzLDgxMy43OVoiLz48cGF0a
CBjbGFzcz0iY2xzLTQiIGQ9Ik04ODkuNzUsOTA4YS4zOS4zOSwwLDAsMS0uMjcuMTFoLS4wNkM4M
DcuMDcsODkzLjkzLDc4Nyw4MTYuODgsNzQzLjA3LDcyM2EzMDcuNDksMzA3LjQ5LDAsMCwwLDIwL
jQ1LTU1LjU0YzExLTQxLjExLDE2LjU5LTkwLjYxLDE2LjU5LTE0Ny4xNCwwLTkzLjA4LTEyLTE3M
y0zNi42Ni0yMzcuNHEtNC4yMi0xMS4xNi04LjIxLjdjODIuNzUsOTAuNTksMTY4LjEyLDIwMS4wNS
wyMDIuNzUsMjQyLjE5QzEwNDQuNzksNjIwLjU2LDEwMDkuMDkuMjcsNzg2Ljg5LDg4NS43NSw5MDhhLi
PC9zdmc+Cg==
...
```

1 base64 값으로 정의된 icon입니다.

웹 콘솔에서 icon 요소 표시

icon은 **Quick Starts** 페이지에서 퀵 스타트의 소개 부분에 표시됩니다.



9.4.2.6. introduction 요소

YAML 파일에서 introduction 요소 표시

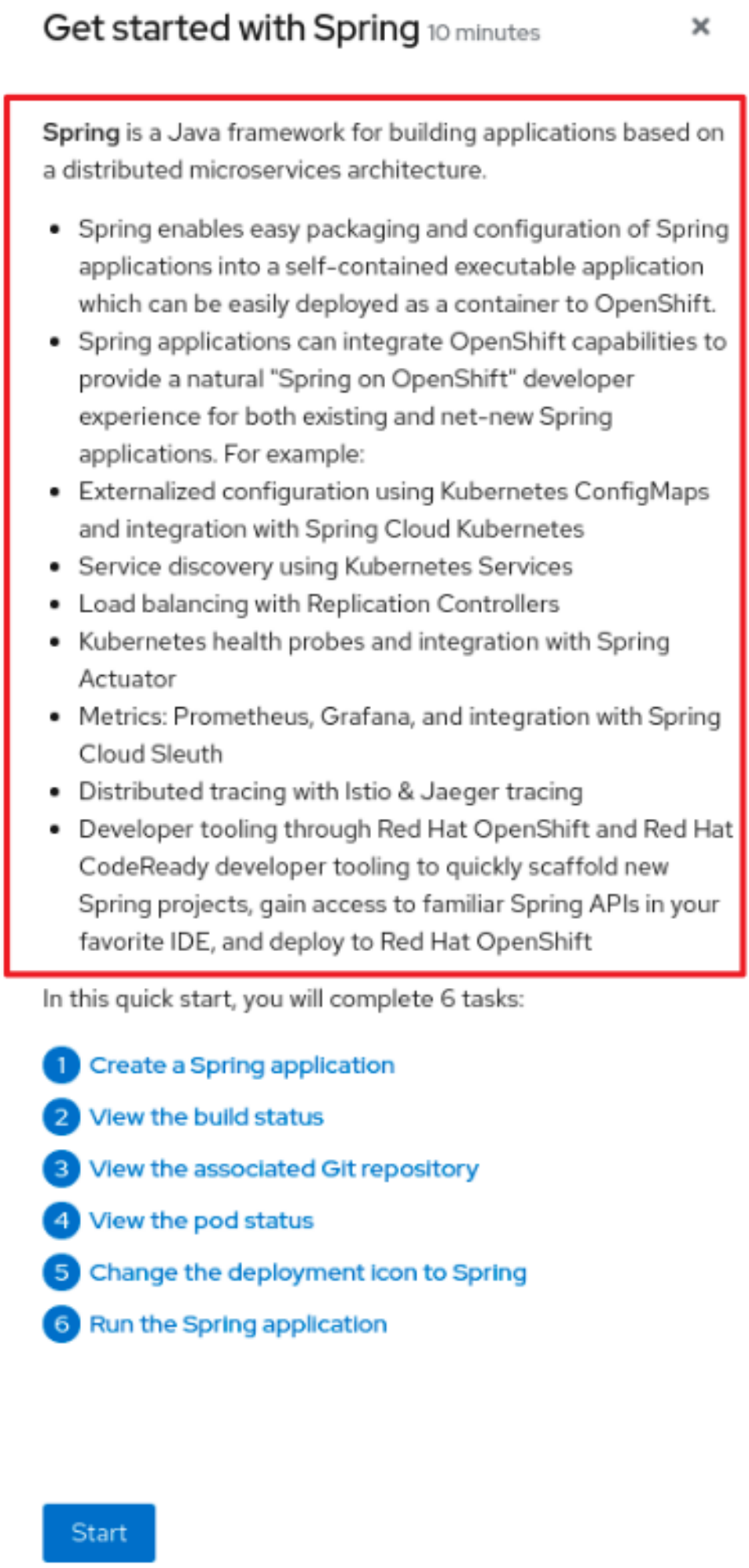
```
...
  introduction: >- 1
    **Spring** is a Java framework for building applications based on a distributed microservices
architecture.

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

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

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

    - Service discovery using Kubernetes Services
```

```
    - Load balancing with Replication Controllers

    - Kubernetes health probes and integration with Spring Actuator

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

    - Distributed tracing with Istio & Jaeger tracing

    - Developer tooling through Red Hat OpenShift and Red Hat CodeReady developer tooling to
quickly scaffold new Spring projects, gain access to familiar Spring APIs in your favorite IDE, and
deploy to Red Hat OpenShift
...
```

1 introduction에서는 퀵 스타트를 소개하고 해당 작업의 목록을 표시합니다.

웹 콘솔에서 introduction 요소 표시

퀵 스타트 카드를 클릭하면 퀵 스타트를 소개하고 퀵 스타트에 포함된 작업 목록이 나열된 사이드 패널이 슬라이딩됩니다.


Get started with Spring 10 minutes
Spring is a Java framework for building applications based on a distributed microservices architecture.
Spring enables easy packaging and configuration of Spring applications into a self-contained executable application which can be easily deployed as a container to OpenShift.
Spring applications can integrate OpenShift capabilities to provide a natural "Spring on OpenShift" developer experience for both existing and net-new Spring applications. For example:
Externalized configuration using Kubernetes ConfigMaps and integration with Spring Cloud Kubernetes
Service discovery using Kubernetes Services
Load balancing with Replication Controllers
Kubernetes health probes and integration with Spring Actuator
Metrics: Prometheus, Grafana, and integration with Spring Cloud Sleuth
Distributed tracing with Istio & Jaeger tracing
Developer tooling through Red Hat OpenShift and Red Hat CodeReady developer tooling to quickly scaffold new Spring projects, gain access to familiar Spring APIs in your favorite IDE, and deploy to Red Hat OpenShift
In this quick start, you will complete 6 tasks:
1 Create a Spring application
2 View the build status
3 View the associated Git repository
4 View the pod status
5 Change the deployment icon to Spring
6 Run the Spring application
Start

9.4.3. 퀵 스타트에 사용자 정의 아이콘 추가

모든 퀵 스타트에 대해 기본 아이콘이 제공됩니다. 사용자 정의 아이콘을 지정할 수 있습니다.

절차

1. 사용자 정의 아이콘으로 사용할 **.svg** 파일을 찾습니다.
2. 온라인 도구를 사용하여 텍스트를 base64로 변환 합니다.
3. YAML 파일에서 **icon: >-**을 추가하고 다음 줄에 **data:image/svg+xml;base64**가 포함되고 그 뒤에 base64 변환의 출력이 포함됩니다. 예를 들면 다음과 같습니다.

```
icon: >-


ciIHJvbGU9ImltZyIgdmlld.
```

9.4.4. 퀵 스타트에 대한 액세스 제한

모든 사용자가 퀵 스타트를 사용할 수 있는 것은 아닙니다. YAML 파일의 **accessReviewResources** 섹션에서는 퀵 스타트에 대한 액세스 제한 기능을 제공합니다.

HelmChartRepository 리소스를 생성하는 기능이 있는 경우에만 사용자가 퀵 스타트에 액세스할 수 있도록 하려면 다음 설정을 사용합니다.

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

Operator 그룹 및 패키지 매니페스트를 나열하고 Operator를 설치할 수 있는 기능이 있는 경우에만 사용자가 퀵 스타트에 액세스할 수 있도록 하려면 다음 설정을 사용합니다.

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

9.4.5. 기타 퀵 스타트 링크

절차

- YAML 파일의 **nextQuickStart** 섹션에서 링크로 연결하려는 퀵 스타트의 **name** (**displayName**이 아님)을 제공합니다. 예를 들면 다음과 같습니다.

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

9.4.6. 퀵 스타트에서 지원되는 태그

이 태그를 사용하여 퀵 스타트 내용을 마크다운으로 작성합니다. 마크다운은 HTML로 변환됩니다.

Tag	설명
'b',	굵은 텍스트를 정의합니다.
'img',	이미지를 포함합니다.
'i',	이탤릭체 텍스트를 정의합니다.
'strike',	취소선 (strike-through) 텍스트를 정의합니다.
's',	작은 텍스트를 정의합니다.
'del',	작은 텍스트를 정의합니다.
'em',	강조된 텍스트를 정의합니다.
'strong',	중요한 텍스트를 정의합니다.
'a',	앵커 태그를 정의합니다.
'p',	단락 텍스트를 정의합니다.
'h1',	레벨 1 제목을 정의합니다.
'h2',	레벨 2 제목을 정의합니다.
'h3',	레벨 3 제목을 정의합니다.
'h4',	레벨 4 제목을 정의합니다.
'ul',	순서가 지정되지 않은 목록을 정의합니다.
'ol',	순서가 지정된 목록을 정의합니다.
'li',	목록 항목을 정의합니다.
'code',	텍스트를 코드로 정의합니다.
'pre',	미리 포맷된 텍스트 블록을 정의합니다.
'button',	텍스트에 버튼을 정의합니다.

9.4.7. 퀵 스타트 강조 표시 참조

강조 표시 또는 힌트 기능을 사용하면 빠른 시작에 웹 콘솔의 구성 요소를 강조 표시하고 예상할 수 있는 링크를 포함할 수 있습니다.

마크다운 구문에는 다음이 포함됩니다.

- 대괄호로 묶은 링크 텍스트
- **highlight** 키워드 다음에 예상 요소의 ID가 표시됨

9.4.7.1. 화면 전환기

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

9.4.7.2. 관리자 화면 탐색 링크

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

9.4.7.3. 개발자 화면 탐색 링크

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

9.4.7.4. 일반적인 탐색 링크

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

9.4.7.5. 마스트 헤드 링크

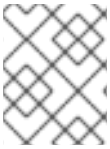
```
[CloudShell]{{highlight qs-masthead-cloudshell}}
[Utility Menu]{{highlight qs-masthead-utilitymenu}}
[User Menu]{{highlight qs-masthead-usermenu}}
[Applications]{{highlight qs-masthead-applications}}
[Import]{{highlight qs-masthead-import}}
[Help]{{highlight qs-masthead-help}}
[Notifications]{{highlight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

9.4.8. 코드 스니펫 마크다운 참조

웹 콘솔에서 CLI 코드 조각이 빠른 시작에 포함된 경우 실행할 수 있습니다. 이 기능을 사용하려면 Web Terminal Operator를 설치해야 합니다. Web Terminal Operator를 설치하지 않는 경우 웹 터미널에서 실행되는 웹 터미널 및 코드 조각 작업이 표시되지 않습니다. 또는 Web Terminal Operator가 설치되어 있는지 여부에 관계없이 코드 조각을 클립보드에 복사할 수 있습니다.

9.4.8.1. 인라인 코드 조각의 구문

```
`code block`{{copy}}
`code block`{{execute}}
```



참고

execute 구문이 사용되는 경우 Web Terminal Operator를 설치했는지 여부에 관계없이 **Copy to clipboard** 작업이 표시됩니다.

9.4.8.2. 여러 줄 코드 조각의 구문

````
```
multi line code block
```{{copy}}

```
multi line code block
```{{execute}}
````

9.5. 퀵 스타트 콘텐츠 가이드 라인

9.5.1. Card copy

퀵 스타트 카드의 제목과 설명을 사용자 지정할 수 있지만 상태를 사용자 정의 할 수 없습니다.

- 설명을 하나 또는 두 문장으로 정리합니다.
- 동사로 시작하고 사용자의 목표를 전달합니다. 올바른 예:

  ```
  Create a serverless application.
  ```

9.5.2. 도입 부분

퀵 스타트 카드를 클릭하면 퀵 스타트를 소개하고 퀵 스타트에 포함된 작업 목록이 나열된 사이드 패널이 슬라이딩됩니다.

- 도입 부분의 내용을 명확하고 간결한 정보로 제공하십시오.
- 퀵 스타트의 결과를 설명하십시오. 사용자가 시작하기 전에 퀵 스타트의 목적을 이해하고 있어야 합니다.
- 퀵 스타트가 아니라 사용자가 수행할 작업을 제공합니다.
  - **올바른 예**:

    ```
    In this quick start, you will deploy a sample application to {product-title}.
    ```

- **잘못된 예**:

> This quick start shows you how to deploy a sample application to {product-title}.

- 도입 부분은 기능의 복잡성에 따라 최대 4~5개의 문장으로 구성해야 합니다. 도입 부분이 긴 경우 사용자를 압도해 버릴 수 있습니다.
- 도입 부분 이후에 퀵 스타트 작업을 나열하고 각 작업의 목록을 동사로 시작합니다. 작업 수를 추가하거나 제거할 때마다 복사본을 업데이트해야 하므로 작업 수를 지정하지 마십시오.
  - **올바른 예**:

> Tasks to complete: Create a serverless application; Connect an event source; Force a new revision

  - **잘못된 예**:

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

9.5.3. 작업 단계

사용자가 **Start**를 클릭하면 퀵 스타트를 완료하기 위해 수행해야 하는 일련의 단계가 표시됩니다.

작업 단계를 작성할 때 다음과 같은 일반적인 지침을 따르십시오.

- 버튼 및 레이블에 대해 "Click"을 사용합니다. 체크 박스, 라디오 버튼, 드롭다운 메뉴에 "Select"를 사용합니다.
- "Click on" 대신 "Click"을 사용합니다.
  - **올바른 예**:

> Click OK.

  - **잘못된 예**:

> Click on the OK button.

- 사용자에게 **Administrator** 및 **Developer** 모드 간의 이동 방법을 보여줍니다. 사용자가 이미 적절한 모드에 있는 경우에도 해당 사용자가 적절한 위치로 이동하는 방법에 대한 지침을 제공합니다. 예:

> Enter the Developer perspective: In the main navigation, click the dropdown menu and select Developer.
> Enter the Administrator perspective: In the main navigation, click the dropdown menu and select Admin.

- "Location, action" 구조를 사용합니다. 사용자에게 수행할 작업을 지시하기 전에 이동해야하는 위치를 알려줍니다.
  - **올바른 예**:

> In the node.js deployment, hover over the icon.

- **잘못된 예**:

  > Hover over the icon in the node.js deployment.

- 제품의 용어에 대해서는 일관되게 대문자를 사용합니다.
- 메뉴 유형이나 목록을 드롭다운으로 지정해야 하는 경우 하이픈 없이 "dropdown"의 한 단어로 작성합니다.
- 사용자 작업과 제품 기능에 대한 추가 정보를 명확히 구분합니다.
  - **사용자 작업**:

    > Change the time range of the dashboard by clicking the dropdown menu and selecting time range.

  - **추가 정보**:

    > To look at data in a specific time frame, you can change the time range of the dashboard.

- "오른쪽 상단 코너에서 아이콘을 클릭" 등과 같은 지시문을 사용하지 마십시오. UI 레이아웃을 변경할 때마다 지시문은 구식이 됩니다. 또한 다른 화면 크기를 가진 사용자에게 데스크탑 사용자의 지침이 정확하지 않을 수 있습니다. 대신 이름을 사용하여 특정 정보를 식별할 수 있도록 합니다.
  - **올바른 예**:

    > In the navigation menu, click Settings.

  - **잘못된 예**:

    > In the left-hand menu, click Settings.

- "회색 원형 클릭"과 같이 색상만으로 항목을 식별하지 마십시오. 색상으로만 항목을 식별하는 것은 시력 제한이 있는사용자, 특히 색맹인 사용자에게는 도움이 되지 않습니다. 대신 버튼 복사와 같이 이름 또는 복사를 사용하여 항목을 식별합니다.
  - **올바른 예**:

    > The success message indicates a connection.

  - **잘못된 예**:

    > The message with a green icon indicates a connection.

- 2 인칭을 일관되게 사용합니다:
  - **올바른 예**:

    > Set up your environment.

  - **잘못된 예**:

    > Let's set up our environment.

9.5.4. 작업 모듈 확인

- 사용자가 단계를 완료하면 **Check your work** 모듈이 표시됩니다. 이 모듈에서는 사용자에게 단계 결과에 대한 질문에 yes 또는 no의 답변을 입력하도록 하므로 사용자는 작업을 검토할 수 있습니다. 이 모듈의 경우, yes 또는 no 의 대답을 요구하는 질문 만 작성해야 합니다.
  - 사용자가 **Yes**로 응답하면 체크 표시가 표시됩니다.
  - 사용자가 **No**로 응답하면 필요에 따라 관련 문서에 대한 링크와 함께 오류 메시지가 표시됩니다. 그러면 사용자가 돌아가서 다시 시도할 수 있습니다.

9.5.5. UI 요소 포맷

다음 지침을 사용하여 UI 요소를 포맷합니다.

- 버튼, 드롭다운, 탭, 필드 및 기타 UI 컨트롤의 복사: UI에 표시되는 대로 복사본을 만들고 이를 굵게 표시합니다.
- 페이지, 창, 패널 이름을 포함한 기타 모든 UI 요소: UI에 표시되는 대로 복사본을 만들고 이를 굵게 표시합니다.
- 코드 또는 사용자 입력 텍스트: 고정폭 글꼴을 사용합니다.
- 힌트: 네비게이션 또는 마스터헤드 요소에 대한 힌트가 포함된 경우 링크와 같이 텍스트 스타일을 지정합니다.
- CLI 명령: 고정폭 글꼴을 사용합니다.
- 실행 중인 텍스트에서 명령에는 굵은 고정폭 글꼴을 사용합니다.
- 매개변수 또는 옵션이 변수 값인 경우, 이텔릭체 고정폭 글꼴을 사용합니다.
- 매개변수에 굵은 고정폭 글꼴을 사용하고 옵션에는 고정폭 글꼴을 사용합니다.

9.6. 추가 리소스

- 어조 및 말투의 요구 사항은 PatternFly's brand voice and tone 가이드 라인 을 참조하십시오.
- 다른 UX 컨텐츠의 지침은 PatternFly's UX writing style guide에서 참조하십시오.