Red Hat Developer Hub 1.2

Configuring plugins in Red Hat Developer Hub

Last Updated: 2024-07-03

Legal Notice

Abstract

Red Hat Developer Hub is a developer platform for building developer portals. You can add and configure plugins in Developer Hub to access various software development tools.

Table of Contents

PREFACE

The Red Hat Developer Hub is an enterprise-grade, integrated developer platform, extended through plugins, that helps reduce the friction and frustration of developers while boosting their productivity.

RED HAT DEVELOPER HUB SUPPORT

If you experience difficulty with a procedure described in this documentation, visit the Red Hat Customer Portal. You can use the Red Hat Customer Portal for the following purposes:

- To search or browse through the Red Hat Knowledgebase of technical support articles about Red Hat products.
- To create a support case for Red Hat Global Support Services (GSS). For support case creation, select **Red Hat Developer Hub**as the product and select the appropriate product version.

CHAPTER 1. PLUGINS IN RED HAT DEVELOPER HUB

The Red Hat Developer Hub application offers a unified platform with various plugins. Using the plugin ecosystem within the Developer Hub application, you can access any kind of development infrastructure or software development tool.

The plugins in Developer Hub maximize the productivity and streamline the development workflows by maintaining the consistency in the overall user experience.

CHAPTER 2. DYNAMIC PLUGIN INSTALLATION

The dynamic plugin support is based on the backend plugin manager package, which is a service that scans a configured root directory (**dynamicPlugins.rootDirectory** in the app config) for dynamic plugin packages and loads them dynamically.

You can use the dynamic plugins that come preinstalled with Red Hat Developer Hub or install external dynamic plugins from a public NPM registry.

2.1. VIEWING INSTALLED PLUGINS

Using the Dynamic Plugins Info front-end plugin, you can view plugins that are currently installed in your Red Hat Developer Hub application. This plugin is enabled by default.

Procedure

1. Open your Developer Hub application and click **Administration**.
2. Go to the **Plugins** tab to view a list of installed plugins and related information.

2.2. PREINSTALLED DYNAMIC PLUGINS

Red Hat Developer Hub is preinstalled with a selection of dynamic plugins. The dynamic plugins that require custom configuration are disabled by default.

For a complete list of dynamic plugins that are preinstalled in this release of Developer Hub, see the *Dynamic plugins support matrix* in the Configuring plugins in Red Hat Developer Hub .

Upon application startup, for each plugin that is disabled by default, the **install-dynamic-plugins init container** within the Developer Hub pod log displays a message similar to the following:

```
======= Skipping disabled dynamic plugin ./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-github-dynamic
```

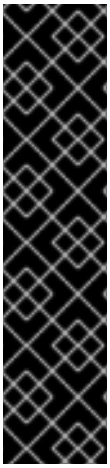
To enable this plugin, add a package with the same name to the Helm chart and change the value in the **disabled** field to 'false'. For example:

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: ./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-github-dynamic
        disabled: false
```

NOTE

The default configuration for a plugin is extracted from the **dynamic-plugins.default.yaml`** file, however, you can use a **pluginConfig** entry to override the default configuration.

2.2.1. Preinstalled dynamic plugin descriptions and details



IMPORTANT

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

For more information on Red Hat Technology Preview features, see Technology Preview Features Scope.

Additional detail on how Red Hat provides support for bundled community dynamic plugins is available on the Red Hat Developer Support Policy page.

There are 60 plugins available in Red Hat Developer Hub. See the following table for more information:

Table 2.1. Dynamic plugins support matrix

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
3scale	@janus-idp/backstage-plugin-3scale-backend	Backend	1.5.13	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-3scale-backend-dynamic	**THREESCALE_BASE_URL** **THREESCALE_ACCESS_TOKEN**	Disabled
AAP	@janus-idp/backstage-plugin-aap-backend	Backend	1.6.13	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-aap-backend-dynamic	**AAP_BASE_URL** **AAP_AUTH_TOKEN**	Disabled
ACR	@janus-idp/backstage-plugin-acr	Frontend	1.4.11	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-acr-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Analytics Provider Segment	@janus-idp/backstage-plugin-analytics-provider-segment	Frontend	1.4.7	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-analytics-provider-segment-dynamic	**SEGMENT_WRITE_KEY** **SEGMENT_TEST_MODE**	Enabled
Argo CD	@janus-idp/backstage-plugin-argocd	Frontend	1.1.6	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-argocd-dynamic		Disabled
Argo CD	@roadiehq/backstage-plugin-argo-cd	Frontend	2.6.5	Production	./dynamic-plugins/dist/roadiehq-backstage-plugin-argo-cd-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Argo CD	@roadiehq/backstage-plugin-argo-cd-backend	Backend	3.0.2	Production	./dynamic-plugins/dist/roadiehq-backstage-plugin-argo-cd-backend-dynamic	**ARGOCD_USERNAME** **ARGOCD_PASSWORD** **ARGOCD_INSTANCE1_URL** **ARGOCD_AUTH_TOKEN** **ARGOCD_INSTANCE2_URL** **ARGOCD_AUTH_TOKEN2**	Disabled
Argo CD	@roadiehq/scaffolder-backend-argocd	Backend	1.1.27	Community Support	./dynamic-plugins/dist/roadiehq-scaffolder-backend-argocd-dynamic	**ARGOCD_USERNAME** **ARGOCD_PASSWORD** **ARGOCD_INSTANCE1_URL** **ARGOCD_AUTH_TOKEN** **ARGOCD_INSTANCE2_URL** **ARGOCD_AUTH_TOKEN2**	Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Azure	@backstage/plugin-scaffolder-backend-module-azure	Backend	0.1.9	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-azure-dynamic		Enabled
Azure Devops	@backstage/plugin-azure-devops	Frontend	0.4.4	Community Support	./dynamic-plugins/dist/backstage-plugin-azure-devops-dynamic		Disabled
Azure Devops	@backstage/plugin-azure-devops-backend	Backend	0.6.5	Community Support	./dynamic-plugins/dist/backstage-plugin-azure-devops-backend-dynamic	**AZURE_TOKEN** **AZURE_ORG**	Disabled
Azure Repositories	@parfuemerie/douglas-scaffolder-backend-module-azure-repositories	Backend	0.2.7	Community Support	./dynamic-plugins/dist/parfuemerie-douglas-scaffolder-backend-module-azure-repositories-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Bitbucket Cloud	@backstage/plugin-catalog-backend-module-bitbucket-cloud	Backend	0.2.4	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-bitbucket-cloud-dynamic	**BITBUCKET_WORKSPACE**	Disabled
Bitbucket Cloud	@backstage/plugin-scaffolder-backend-module-bitbucket-cloud	Backend	0.1.7	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-bitbucket-cloud-dynamic		Enabled
Bitbucket Server	@backstage/plugin-catalog-backend-module-bitbucket-server	Backend	0.1.31	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-bitbucket-server-dynamic	**BITBUCKET_HOST**	Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Bitbucket Server	@backstage/plugin-scaffolder-backend-module-bitbucket-server	Backend	0.1.7	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-bitbucket-server-dynamic		Enabled
Datadog	@roadiehq/backstage-plugin-datadog	Frontend	2.2.8	Community Support	./dynamic-plugins/dist/roadiehq-backstage-plugin-datadog-dynamic		Disabled
Dynatrace	@backstage/plugin-dynatrace	Frontend	10.0.4	Community Support	./dynamic-plugins/dist/backstage-plugin-dynatrace-dynamic		Disabled
Gerrit	@backstage/plugin-scaffolder-backend-module-gerrit	Backend	0.1.9	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-gerrit-dynamic		Enabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
GitHub	@backstage/plugin-catalog-backend-module-github	Backend	0.6.0	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-github-dynamic	**GITHUB_ORG**	Disabled
GitHub	@backstage/plugin-scaffolder-backend-module-github	Backend	0.2.7	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-github-dynamic		Enabled
GitHub Actions	@backstage/plugin-github-actions	Frontend	0.6.16	Community Support	./dynamic-plugins/dist/backstage-plugin-github-actions-dynamic		Disabled
GitHub Insights	@roadiehq/backstage-plugin-github-insights	Frontend	2.3.29	Community Support	./dynamic-plugins/dist/roadiehq-backstage-plugin-github-insights-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
GitHub Issues	@backstage/plugin-github-issues	Frontend	0.4.2	Community Support	./dynamic-plugins/dist/backstage-plugin-github-issues-dynamic		Disabled
GitHub Org	@backstage/plugin-catalog-backend-module-github-org	Backend	0.1.12	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-github-org-dynamic	**GITHUB_URL** **GITHUB_ORG**	Disabled
GitHub Pull Requests	@roadiehq/backstage-plugin-github-pull-requests	Frontend	2.5.26	Community Support	./dynamic-plugins/dist/roadiehq-backstage-plugin-github-pull-requests-dynamic		Disabled
GitLab	@immobiliarelabs/backstage-plugin-gitlab	Frontend	6.5.0	Community Support	./dynamic-plugins/dist/immobiliarelabs-backstage-plugin-gitlab-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
GitLab	@backstage/plugin-catalog-backend-module-gitlab	Backend	0.3.15	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-gitlab-dynamic		Disabled
GitLab	@immobiliarelabs/backstage-plugin-gitlab-backend	Backend	6.5.0	Community Support	./dynamic-plugins/dist/immobiliarelabs-backstage-plugin-gitlab-backend-dynamic	**GITLAB_HOST** **GITLAB_TOKEN**	Disabled
GitLab	@backstage/plugin-scaffolder-backend-module-gitlab	Backend	0.3.3	Community Support	./dynamic-plugins/dist/backstage-plugin-scaffolder-backend-module-gitlab-dynamic		Enabled
GitLab Org	@backstage/plugin-catalog-backend-module-gitlab-org	Backend	0.3.10	Community Support	./dynamic-plugins/dist/backstage-plugin-catalog-backend-module-gitlab-org-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Http Request	@roadiehq/scaffolder-backend-module-http-request	Backend	4.3.2	Community Support	./dynamic-plugins/dist/roadiehq-scaffolder-backend-module-http-request-dynamic		Enabled
Jenkins	@backstage/plugin-jenkins	Frontend	0.9.10	Community Support	./dynamic-plugins/dist/backstage-plugin-jenkins-dynamic		Disabled
Jenkins	@backstage/plugin-jenkins-backend	Backend	0.4.5	Community Support	./dynamic-plugins/dist/backstage-plugin-jenkins-backend-dynamic	**JENKINS_URL** **JENKINS_USERNAME** **JENKINS_TOKEN**	Disabled
JFrog Artifactory	@janus-idp/backstage-plugin-jfrog-artifactory	Frontend	1.4.9	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-jfrog-artifactory-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Jira	@roadiehq/backstage-plugin-jira	Frontend	2.5.8	Community Support	./dynamic-plugins/dist/roadiehq-backstage-plugin-jira-dynamic		Disabled
Keycloak	@janus-idp/backstage-plugin-keycloak-backend	Backend	1.9.10	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-keycloak-backend-dynamic	**KEYCLOAK_BASE_URL** **KEYCLOAK_LOGIN_REALM** **KEYCLOAK_REALM** **KEYCLOAK_CLIENT_ID** **KEYCLOAK_CLIENT_SECRET**	Disabled
Kubernetes	@backstage/plugin-kubernetes	Frontend	0.11.9	Community Support	./dynamic-plugins/dist/backstage-plugin-kubernetes-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Kubernetes	@backstage/plugin-kubernetes-backend	Backend	0.17.0	Production	./dynamic-plugins/dist/backstage-plugin-kubernetes-backend-dynamic	**K8S_CLUSTER_NAME** **K8S_CLUSTER_URL** **K8S_CLUSTER_TOKEN**	Disabled
Lighthouse	@backstage/plugin-lighthouse	Frontend	0.4.20	Community Support	./dynamic-plugins/dist/backstage-plugin-lighthouse-dynamic		Disabled
Nexus Repository Manager	@janus-idp/backstage-plugin-nexus-repository-manager	Frontend	1.6.8	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-nexus-repository-manager-dynamic		Disabled
OCM	@janus-idp/backstage-plugin-ocm	Frontend	4.1.6	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-ocm-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
OCM	@janus-idp/backstage-plugin-ocm-backend	Backend	4.0.6	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-ocm-backend-dynamic	**OCM_HUB_NAME** **OCM_HUB_URL** **moc_infra_token**	Disabled
PagerDuty	@pagerduty/backstage-plugin	Frontend	0.12.0	Community Support	./dynamic-plugins/dist/pagerduty-backstage-plugin-dynamic		Disabled
Quay	@janus-idp/backstage-plugin-quay	Frontend	1.7.6	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-quay-dynamic		Disabled
Quay	@janus-idp/backstage-scaffolder-backend-module-quay	Backend	1.4.10	Production	./dynamic-plugins/dist/janus-idp-backstage-scaffolder-backend-module-quay-dynamic		Enabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
RBAC	@janus-idp/backstage-plugin-rbac	Frontend	1.20.11	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-rbac-dynamic		Disabled
Regex	@janus-idp/backstage-scaffolder-backend-module-regex	Backend	1.4.10	Production	./dynamic-plugins/dist/janus-idp-backstage-scaffolder-backend-module-regex-dynamic		Enabled
Scaffolder Relation Processor	@janus-idp/backstage-plugin-catalog-backend-module-scaffolder-relation-processor	Backend	1.0.1	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-plugin-catalog-backend-module-scaffolder-relation-processor-dynamic		Enabled
Security Insights	@roadiehq/backstage-plugin-security-insights	Frontend	2.3.17	Community Support	./dynamic-plugins/dist/roadiehq-backstage-plugin-security-insights-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
ServiceNow	@janus-idp/backstage-scaffolder-backend-module-servicenow	Backend	1.4.12	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-scaffolder-backend-module-servicenow-dynamic	**SERVICENOW_BASE_URL** **SERVICENOW_USERNAME** **SERVICENOW_PASSWORD**	Disabled
SonarQube	@backstage/plugin-sonarqube	Frontend	0.7.17	Community Support	./dynamic-plugins/dist/backstage-plugin-sonarqube-dynamic		Disabled
SonarQube	@backstage/plugin-sonarqube-backend	Backend	0.2.20	Community Support	./dynamic-plugins/dist/backstage-plugin-sonarqube-backend-dynamic	**SONARQUBE_URL** **SONARQUBE_TOKEN**	Disabled
SonarQube	@janus-idp/backstage-scaffolder-backend-module-sonarqube	Backend	1.4.10	Red Hat Tech Preview	./dynamic-plugins/dist/janus-idp-backstage-scaffolder-backend-module-sonarqube-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
TechDocs	@backstage/plugin-techdocs	Frontend	1.10.4	Production	./dynamic-plugins/dist/backstage-plugin-techdocs-dynamic		Enabled
TechDocs	@backstage/plugin-techdocs-backend	Backend	1.10.4	Production	./dynamic-plugins/dist/backstage-plugin-techdocs-backend-dynamic		Enabled
Tech Radar	@backstage/plugin-tech-radar	Frontend	0.7.4	Community Support	./dynamic-plugins/dist/backstage-plugin-tech-radar-dynamic		Disabled
Tekton	@janus-idp/backstage-plugin-tekton	Frontend	3.7.5	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-tekton-dynamic		Disabled
Topology	@janus-idp/backstage-plugin-topology	Frontend	1.21.7	Production	./dynamic-plugins/dist/janus-idp-backstage-plugin-topology-dynamic		Disabled

Name	Plugin	Role	Version	Support Level	Path	Required Variables	Default
Utils	@roadiehq/scaffolder-backend-module-utils	Backend	1.15.3	Community Support	./dynamic-plugins/dist/roadiehq-scaffolder-backend-module-utils-dynamic		Enabled

NOTE

- To configure Keycloak, see Installation and configuration of Keycloak.
- To configure Techdocs, see reference documentation. After experimenting with basic setup, use CI/CD to generate docs and an external cloud storage when deploying TechDocs for production use-case. See also this recommended deployment approach.

2.3. INSTALLATION OF DYNAMIC PLUGINS USING THE HELM CHART

You can deploy a Developer Hub instance using a Helm chart, which is a flexible installation method. With the Helm chart, you can sideload dynamic plugins into your Developer Hub instance without having to recompile your code or rebuild the container.

To install dynamic plugins in Developer Hub using Helm, add the following **global.dynamic** parameters in your Helm chart:

- **plugins**: the dynamic plugins list intended for installation. By default, the list is empty. You can populate the plugins list with the following fields:
  - **package**: a package specification for the dynamic plugin package that you want to install. You can use a package for either a local or an external dynamic plugin installation. For a local installation, use a path to the local folder containing the dynamic plugin. For an external installation, use a package specification from a public NPM repository.
  - **integrity** (required for external packages): an integrity checksum in the form of **<alg>-<digest>** specific to the package. Supported algorithms include **sha256**, **sha384** and **sha512**.
  - **pluginConfig**: an optional plugin-specific **app-config** YAML fragment. See plugin configuration for more information.
  - **disabled**: disables the dynamic plugin if set to **true**. Default: **false**.
- **includes**: a list of YAML files utilizing the same syntax.

NOTE

The **plugins** list in the **includes** file is merged with the **plugins** list in the main Helm values. If a plugin package is mentioned in both **plugins** lists, the **plugins** fields in the main Helm values override the **plugins** fields in the **includes** file. The default configuration includes the **dynamic-plugins.default.yaml** file, which contains all of the dynamic plugins preinstalled in Developer Hub, whether enabled or disabled by default.

2.3.1. Obtaining the integrity checksum

To obtain the integrity checksum, enter the following command:

```
npm view <package name>@<version> dist.integrity
```

2.3.2. Example Helm chart configurations for dynamic plugin installations

The following examples demonstrate how to configure the Helm chart for specific types of dynamic plugin installations.

Configuring a local plugin and an external plugin when the external plugin requires a specific app-config

```
global:
  dynamic:
    plugins:
      - package: <alocal package-spec used by npm pack>
      - package: <external package-spec used by npm pack>
        integrity: sha512-<some hash>
        pluginConfig: ...
```

Disabling a plugin from an included file

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: <some imported plugins listed in dynamic-plugins.default.yaml>
        disabled: true
```

Enabling a plugin from an included file

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: <some imported plugins listed in dynamic-plugins.custom.yaml>
        disabled: false
```

Enabling a plugin that is disabled in an included file

```
global:
```

```
dynamic:
  includes:
    - dynamic-plugins.default.yaml
  plugins:
    - package: <some imported plugins listed in dynamic-plugins.custom.yaml>
      disabled: false
```

2.3.3. Installing external dynamic plugins using a Helm chart

The NPM registry contains external dynamic plugins that you can use for demonstration purposes. For example, the following community plugins are available in the **janus-idp** organization in the NPMJS repository:

- Notifications (frontend and backend)
- Kubernetes actions (scaffolder actions)

To install the Notifications and Kubernetes actions plugins, include them in the Helm chart values in the **global.dynamic.plugins** list as shown in the following example:

```
global:
  dynamic:
    plugins:
      - package: '@janus-idp/plugin-notifications-backend-dynamic@1.3.6'
        # Integrity can be found at https://registry.npmjs.org/@janus-idp/plugin-notifications-backend-
dynamic
        integrity: 'sha512-
Qd8pniy1yRx+x7LnwjzQ6k9zP+C1yex24MaCcx7dGDPT/XbTokwoSZr4baSSn8jUA6P45NUUevu1d629
mG4JGQ=='
      - package: '@janus-idp/plugin-notifications@1.1.12
'
        # https://registry.npmjs.org/@janus-idp/plugin-notifications

        integrity: 'sha512-
GCdEuHRQek3ay428C8C4wWgxjNpNwCXgIdFbUUFGCLLkBFSaOEw+XaBvWaBGtQ5BLgE3jQEUx
a+422uzSYC5oQ=='
        pluginConfig:
          dynamicPlugins:
            frontend:
              janus-idp.backstage-plugin-notifications:
                appIcons:
                  - name: notificationsIcon
                    module: NotificationsPlugin
                    importName: NotificationsActiveIcon
                dynamicRoutes:
                  - path: /notifications
                    importName: NotificationsPage
                    module: NotificationsPlugin
                    menuItem:
                      icon: notificationsIcon
                      text: Notifications
                    config:
                      pollingIntervalMs: 5000
      - package: '@janus-idp/backstage-scaffolder-backend-module-kubernetes-dynamic@1.3.5'
        # https://registry.npmjs.org/@janus-idp/backstage-scaffolder-backend-module-kubernetes-
```

```
dynamic
        integrity: 'sha512-
19ie+FM3QHxWYPyYzE0uNdI5K8M4vGZ0SPeeTw85XPROY1DrIY7rMm2G0XT85L0ZmntHVwc9qW
+SbHolPg/qRA=='
          proxy:
            endpoints:
              /explore-backend-completed:
                target: 'http://localhost:7017'
      - package: '@dfatwork-pkgs/search-backend-module-explore-wrapped-dynamic@0.1.3-next.1'
        # https://registry.npmjs.org/@dfatwork-pkgs/search-backend-module-explore-wrapped-dynamic
        integrity: 'sha512-
mv6LS8UOve+eumoMCVypGcd7b/L36lH2z11tGKVrt+m65VzQI4FgAJr9kNCrjUZPMyh36KVGIjYqsu9+
kgzH5A=='
      - package: '@dfatwork-pkgs/plugin-catalog-backend-module-test-dynamic@0.0.0'
        # https://registry.npmjs.org/@dfatwork-pkgs/plugin-catalog-backend-module-test-dynamic
        integrity: 'sha512-
YsrZMThxJk7cYJU9FtAcsTCx9lCChpytK254TfGb3iMAYQyVcZnr5AA/AU+hezFnXLsr6gj8PP7z/mCZie
uuDA=='
```

2.4. INSTALLING EXTERNAL PLUGINS IN AN AIR-GAPPED ENVIRONMENT

You can install external plugins in an air-gapped environment by setting up a custom NPM registry. To configure the NPM registry URL and authentication information for dynamic plugin packages, see Using a custom NPM registry for dynamic plugin packages.

2.5. USING A CUSTOM NPM REGISTRY FOR DYNAMIC PLUGIN PACKAGES

You can configure the NPM registry URL and authentication information for dynamic plugin packages using a Helm chart. For dynamic plugin packages obtained through **npm pack**, you can use a **.npmrc** file.

Using the Helm chart, add the **.npmrc** file to the NPM registry by creating a secret named **dynamic-plugins-npmrc** with the following content:

```
apiVersion: v1
kind: Secret
metadata:
  name: dynamic-plugins-npmrc
type: Opaque
stringData:
  .npmrc: |
    registry=<registry-url>
    //<registry-url>:_authToken=<auth-token>
          ...
```

2.6. BASIC CONFIGURATION OF DYNAMIC PLUGINS

Some dynamic plugins require environment variables to be set. If a mandatory environment variable is not set, and the plugin is enabled, then the application might fail at startup.

The mandatory environment variables for each plugin are listed in the Dynamic plugins support matrix.

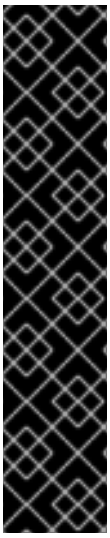
NOTE

Zib-bomb detection When installing some dynamic plugin containing large files, if the installation script considers the package archive to be a Zib-Bomb, the installation fails.

To increase the maximum permitted size of a file inside a package archive, you can increase the **MAX_ENTRY_SIZE** environment value of the deployment **install-dynamic-plugins initContainer** from the default size of **20000000** bytes.

2.7. INSTALLATION AND CONFIGURATION OF ANSIBLE AUTOMATION PLATFORM

The Ansible Automation Platform (AAP) plugin synchronizes the accessible templates including job templates and workflow job templates from AAP into your Developer Hub catalog.



IMPORTANT

The Ansible Automation Platform plugin is a Technology Preview feature only.

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

For more information on Red Hat Technology Preview features, see Technology Preview Features Scope.

Additional detail on how Red Hat provides support for bundled community dynamic plugins is available on the Red Hat Developer Support Policy page.

2.7.1. For administrators

2.7.1.1. Installing and configuring the AAP Backend plugin

The AAP backend plugin allows you to configure one or multiple providers using your **app-config.yaml** configuration file in Developer Hub.

Prerequisites

- Your Developer Hub application is installed and running.
- You have created an account in Ansible Automation Platform.

Installation

The AAP backend plugin is pre-loaded in Developer Hub with basic configuration properties. To enable it, set the **disabled** property to **false** as follows:

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
```

```
plugins:
  - package: ./dynamic-plugins/dist/janus-idp-backstage-plugin-aap-backend-dynamic
    disabled: false
```

Basic configuration

To enable the AAP plugin, you must set the following environment variables:

- **AAP_BASE_URL**: Base URL of the service
- **AAP AUTH TOKEN**: Authentication token for the service

Advanced configuration

1. You can use the **aap** marker to configure the **app-config.yaml** file of Developer Hub as follows:

```
catalog:
  providers:
    aap:
      dev:
        baseUrl: $(AAP_BASE_URL)
        authorization: 'Bearer ${AAP_AUTH_TOKEN}'
        owner: <owner>
        system: <system>
        schedule: # optional; same options as in TaskScheduleDefinition
          # supports cron, ISO duration, "human duration" as used in code
          frequency: { minutes: 1 }
          # supports ISO duration, "human duration" as used in code
          timeout: { minutes: 1 }
```

2.7.1.2. Log lines for AAP Backend plugin troubleshoot

When you start your Developer Hub application, you can see the following log lines:

```
[1] 2023-02-13T15:26:09.356Z catalog info Discovered ResourceEntity API type=plugin
target=AapResourceEntityProvider:dev
[1] 2023-02-13T15:26:09.423Z catalog info Discovered ResourceEntity Red Hat Event (DEV, v1.2.0)
type=plugin target=AapResourceEntityProvider:dev
[1] 2023-02-13T15:26:09.620Z catalog info Discovered ResourceEntity Red Hat Event (TEST, v1.1.1)
type=plugin target=AapResourceEntityProvider:dev
[1] 2023-02-13T15:26:09.819Z catalog info Discovered ResourceEntity Red Hat Event (PROD,
v1.1.1) type=plugin target=AapResourceEntityProvider:dev
[1] 2023-02-13T15:26:09.819Z catalog info Applying the mutation with 3 entities type=plugin
target=AapResourceEntityProvider:dev
```

2.7.2. For users

2.7.2.1. Accessing templates from AAP in Developer Hub

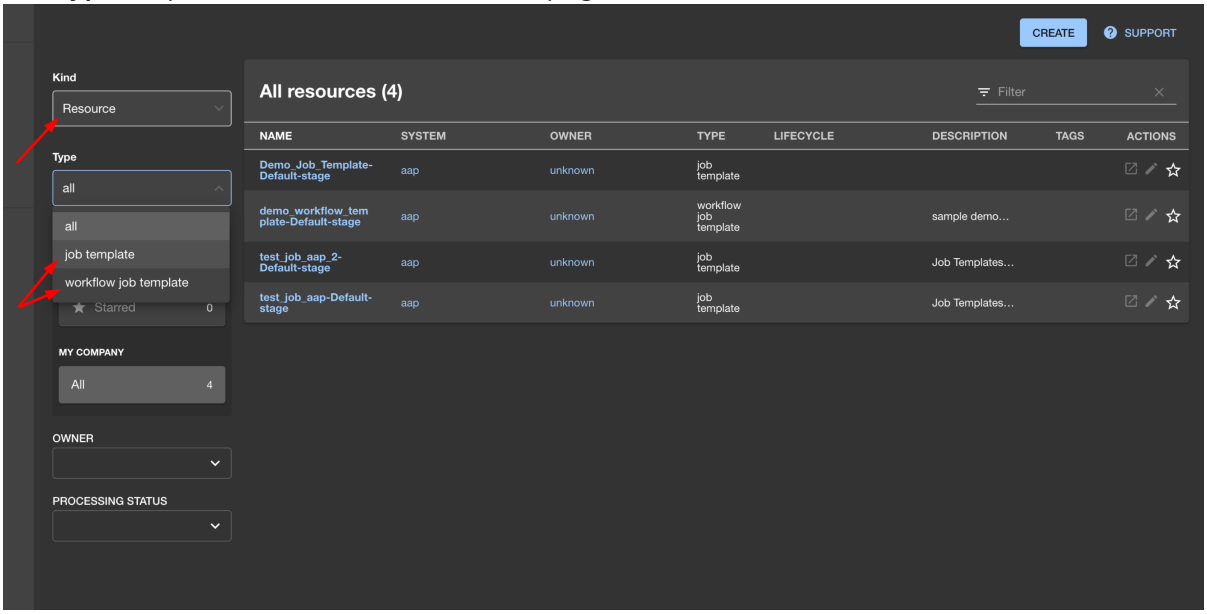
When you have configured the AAP backend plugin successfully, it synchronizes the templates including job templates and workflow job templates from AAP and displays them on the Developer Hub Catalog page as Resources.

Prerequisites

- Your Developer Hub application is installed and running.
- You have installed the AAP backend plugin. For installation and configuration instructions, see Section 2.7.1.1, “Installing and configuring the AAP Backend plugin” .

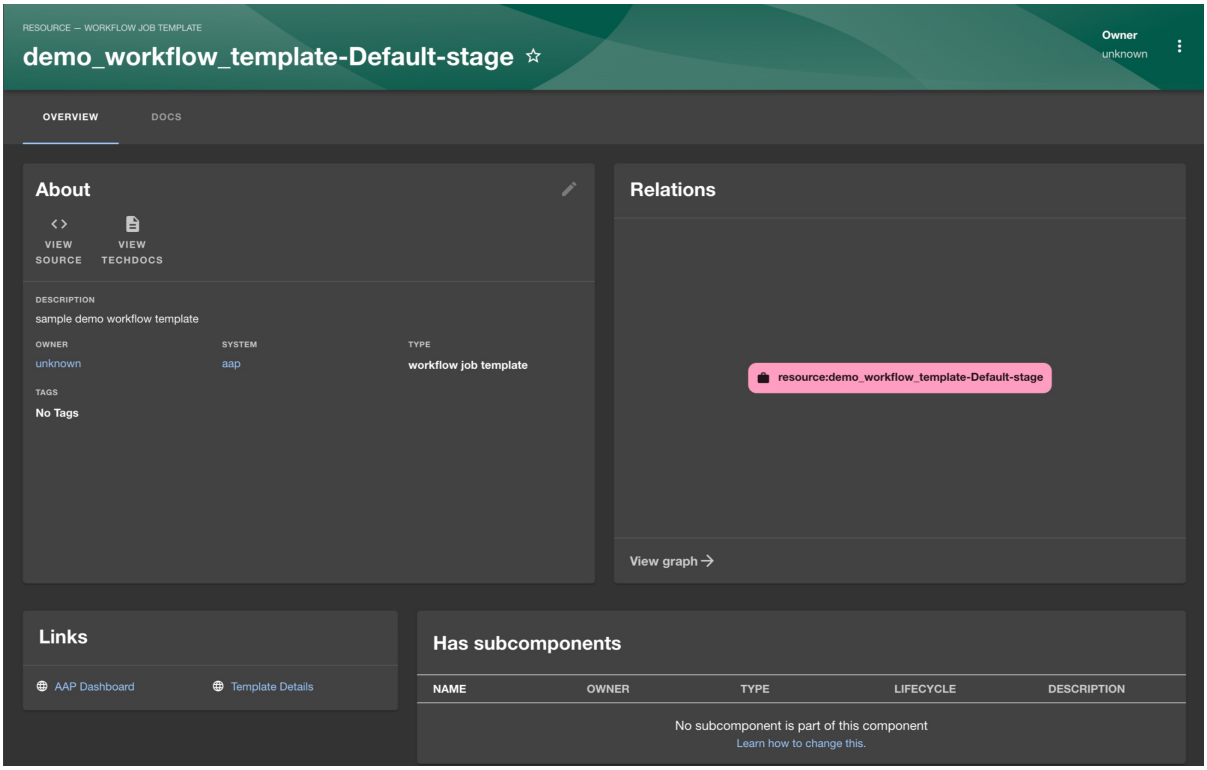
Procedure

1. Open your Developer Hub application and Go to the **Catalog** page.
2. Select **Resource** from the **Kind** drop-down and **job template** or **workflow job template** from the **Type** drop-down on the left side of the page.



A list of all the available templates from AAP appears on the page.

3. Select a template from the list.
The **OVERVIEW** tab appears containing different cards, such as:
   - **About**: Provides detailed information about the template.
   - **Relations**: Displays the visual representation of the template and associated aspects.
   - **Links**: Contains links to the AAP dashboard and the details page of the template.
   - **Has subcomponents**: Displays a list of associated subcomponents.



2.8. INSTALLATION AND CONFIGURATION OF KEYCLOAK

The Keycloak backend plugin, which integrates Keycloak into Developer Hub, has the following capabilities:

- Synchronization of Keycloak users in a realm.
- Synchronization of Keycloak groups and their users in a realm.

2.8.1. For administrators

2.8.1.1. Installation

The Keycloak plugin is pre-loaded in Developer Hub with basic configuration properties. To enable it, set the **disabled** property to **false** as follows:

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: ./dynamic-plugins/dist/janus-idp-backstage-plugin-keycloak-backend-dynamic
        disabled: false
```

2.8.1.2. Basic configuration

To enable the Keycloak plugin, you must set the following environment variables:

- **KEYCLOAK_BASE_URL**
- **KEYCLOAK_LOGIN_REALM**

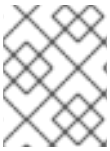
- **KEYCLOAK_REALM**
- **KEYCLOAK_CLIENT_ID**
- **KEYCLOAK_CLIENT_SECRET**

2.8.1.3. Advanced configuration

Schedule configuration

You can configure a schedule in the **app-config.yaml** file, as follows:

```
catalog:
  providers:
    keycloakOrg:
      default:
        # ...
        # highlight-add-start
        schedule: # optional; same options as in TaskScheduleDefinition
          # supports cron, ISO duration, "human duration" as used in code
          frequency: { minutes: 1 }
          # supports ISO duration, "human duration" as used in code
          timeout: { minutes: 1 }
          initialDelay: { seconds: 15 }
          # highlight-add-end
```



NOTE

If you have made any changes to the schedule in the **app-config.yaml** file, then restart to apply the changes.

Keycloak query parameters

You can override the default Keycloak query parameters in the **app-config.yaml** file, as follows:

```
catalog:
  providers:
    keycloakOrg:
      default:
        # ...
        # highlight-add-start
        userQuerySize: 500 # Optional
        groupQuerySize: 250 # Optional
        # highlight-add-end
```

Communication between Developer Hub and Keycloak is enabled by using the Keycloak API. Username and password, or client credentials are supported authentication methods.

The following table describes the parameters that you can configure to enable the plugin under **catalog.providers.keycloakOrg.<ENVIRONMENT_NAME>** object in the **app-config.yaml** file:

Name	Description	Default Value	Required
baseUrl	Location of the Keycloak server, such as **https://localhost:8443/auth**. Note that the newer versions of Keycloak omit the **/auth** context path.	""	Yes
realm	Realm to synchronize	**master**	No
loginRealm	Realm used to authenticate	**master**	No
username	Username to authenticate	""	Yes if using password based authentication
password	Password to authenticate	""	Yes if using password based authentication
clientId	Client ID to authenticate	""	Yes if using client credentials based authentication
clientSecret	Client Secret to authenticate	""	Yes if using client credentials based authentication
userQuerySize	Number of users to query at a time	**100**	No
groupQuerySize	Number of groups to query at a time	**100**	No

When using client credentials, the access type must be set to **confidential** and service accounts must be enabled. You must also add the following roles from the **realm-management** client role:

- **query-groups**
- **query-users**
- **view-users**

2.8.1.4. Limitations

If you have self-signed or corporate certificate issues, you can set the following environment variable before starting Developer Hub:

NODE_TLS_REJECT_UNAUTHORIZED=0

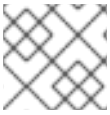
NOTE

The solution of setting the environment variable is not recommended.

2.8.2. For users

2.8.2.1. Import of users and groups in Developer Hub using the Keycloak plugin

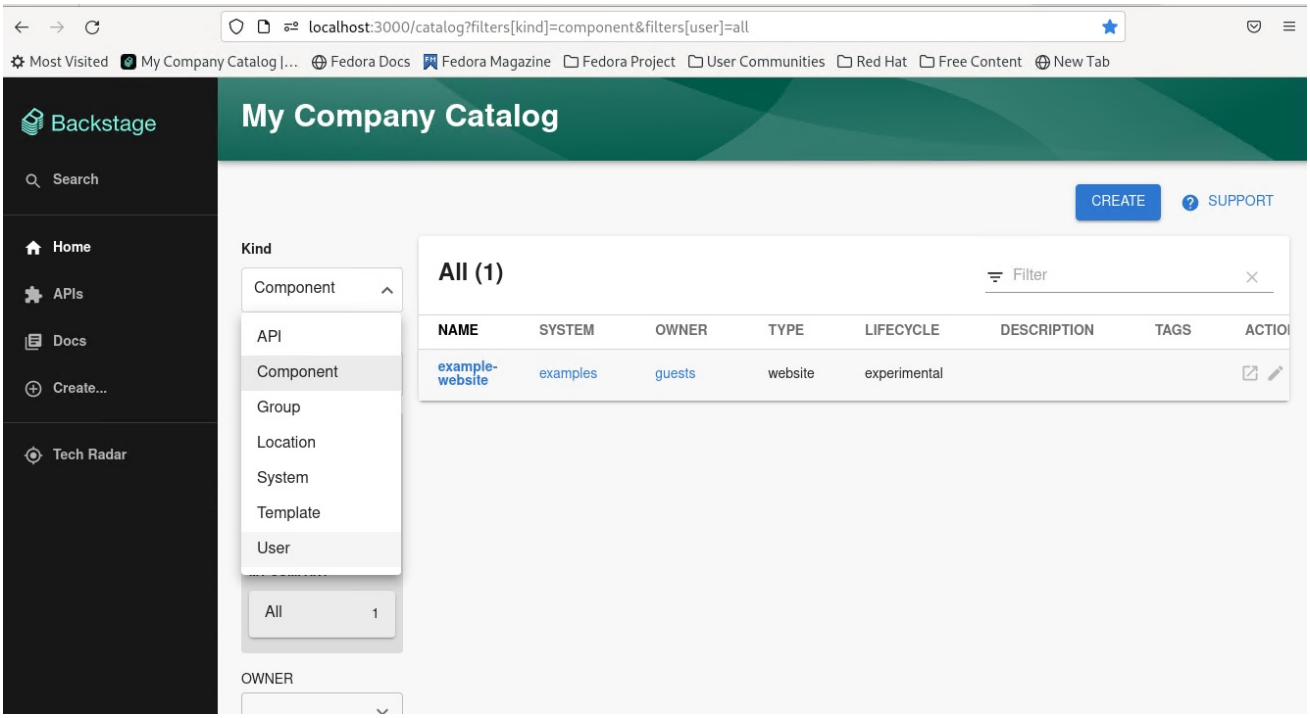
After configuring the plugin successfully, the plugin imports the users and groups each time when started.



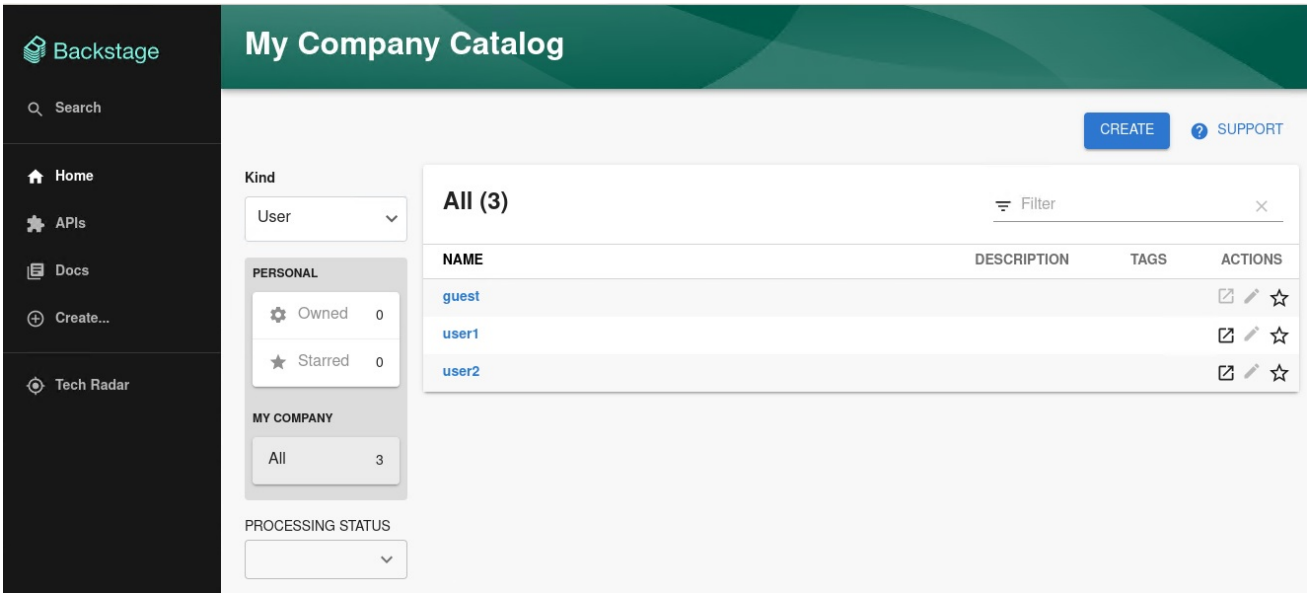
NOTE

If you set up a schedule, users and groups will also be imported.

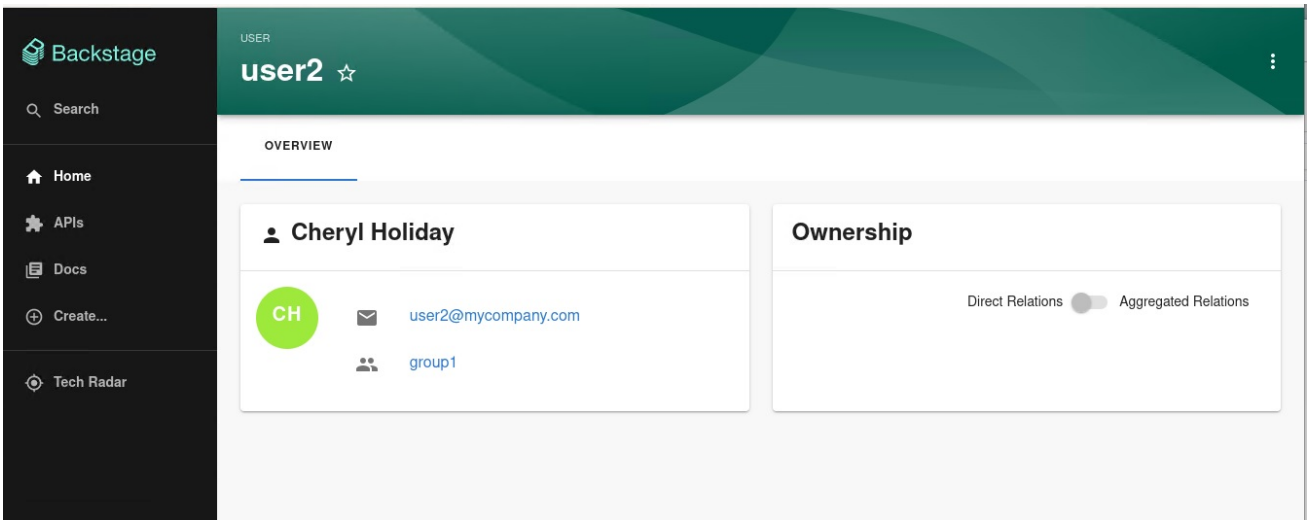
After the first import is complete, you can select **User** to list the users from the catalog page:



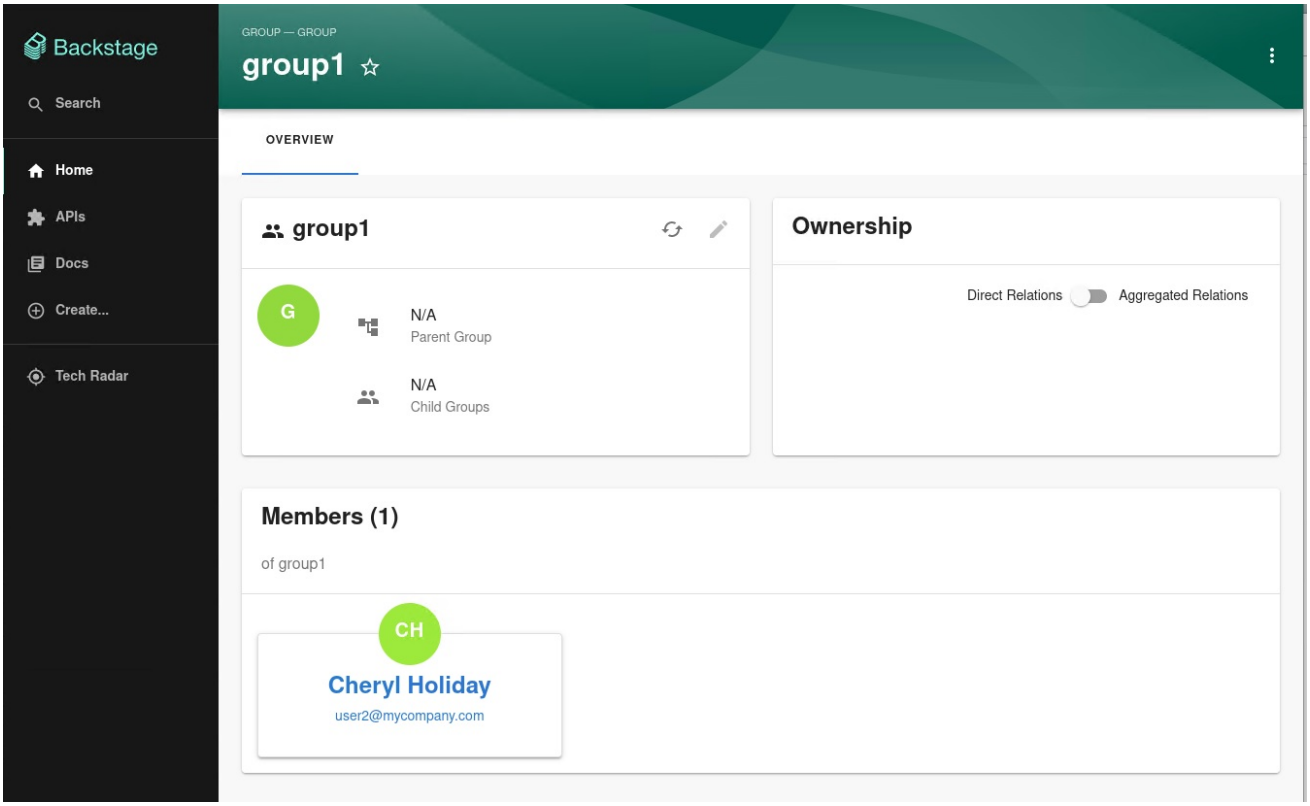
You can see the list of users on the page:



When you select a user, you can see the information imported from Keycloak:



You can also select a group, view the list, and select or view the information imported from Keycloak for a group:



2.9. INSTALLATION AND CONFIGURATION OF NEXUS REPOSITORY MANAGER

The Nexus Repository Manager plugin displays the information about your build artifacts in your Developer Hub application. The build artifacts are available in the Nexus Repository Manager.

IMPORTANT

The Nexus Repository Manager plugin is a Technology Preview feature only.

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

For more information on Red Hat Technology Preview features, see Technology Preview Features Scope.

Additional detail on how Red Hat provides support for bundled community dynamic plugins is available on the Red Hat Developer Support Policy page.

2.9.1. For administrators

2.9.1.1. Installing and configuring the Nexus Repository Manager plugin

Installation

The Nexus Repository Manager plugin is pre-loaded in Developer Hub with basic configuration properties. To enable it, set the disabled property to **false** as follows:

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: ./dynamic-plugins/dist/janus-idp-backstage-plugin-nexus-repository-manager
        disabled: false
```

Configuration

1. Set the proxy to the desired Nexus Repository Manager server in the **app-config.yaml** file as follows:

```
proxy:
  '/nexus-repository-manager':
  target: 'https://<NEXUS_REPOSITORY_MANAGER_URL>'
  headers:
    X-Requested-With: 'XMLHttpRequest'
    # Uncomment the following line to access a private Nexus Repository Manager using a token
    # Authorization: 'Bearer <YOUR TOKEN>'
  changeOrigin: true
  # Change to "false" in case of using self hosted Nexus Repository Manager instance with a self-signed certificate
  secure: true
```

2. Optional: Change the base URL of Nexus Repository Manager proxy as follows:

```
nexusRepositoryManager:
  # default path is `/nexus-repository-manager`
  proxyPath: /custom-path
```

3. Optional: Enable the following experimental annotations:

```
nexusRepositoryManager:
  experimentalAnnotations: true
```

4. Annotate your entity using the following annotations:

```
metadata:
  annotations:
  # insert the chosen annotations here
  # example
  nexus-repository-manager/docker.image-name: `<ORGANIZATION>/<REPOSITORY>`,
```

2.9.2. For users

2.9.2.1. Using the Nexus Repository Manager plugin in Developer Hub

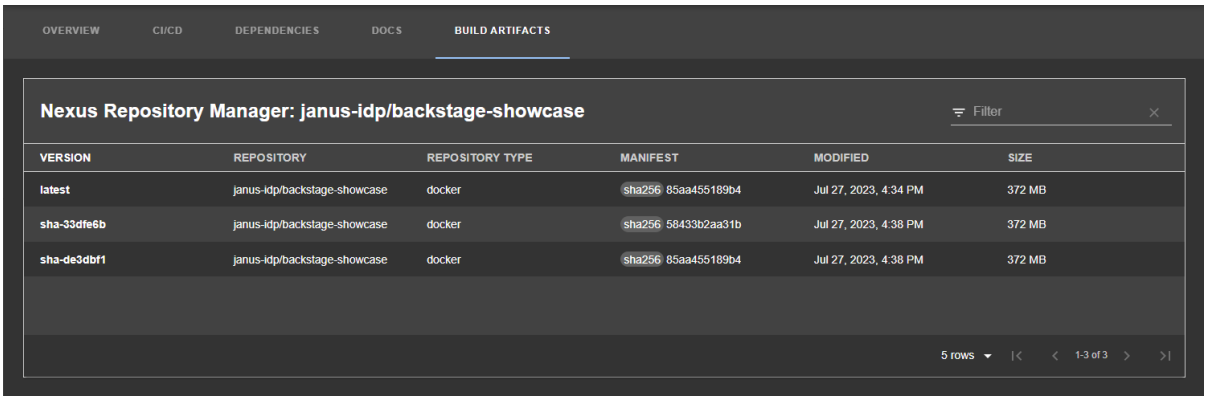
The Nexus Repository Manager is a front-end plugin that enables you to view the information about build artifacts.

Prerequisites

- Your Developer Hub application is installed and running.
- You have installed the Nexus Repository Manager plugin. For the installation process, see Section 2.9.1.1, “Installing and configuring the Nexus Repository Manager plugin” .

Procedure

1. Open your Developer Hub application and select a component from the **Catalog** page.
2. Go to the **BUILD ARTIFACTS** tab.
   The **BUILD ARTIFACTS** tab contains a list of build artifacts and related information, such as **VERSION**, **REPOSITORY**, **REPOSITORY TYPE**, **MANIFEST**, **MODIFIED**, and **SIZE**.



2.10. INSTALLATION AND CONFIGURATION OF TEKTON

You can use the Tekton plugin to visualize the results of CI/CD pipeline runs on your Kubernetes or OpenShift clusters. The plugin allows users to visually see high level status of all associated tasks in the pipeline for their applications.

2.10.1. For administrators

2.10.1.1. Installation

Prerequsites

- You have installed and configured the **@backstage/plugin-kubernetes** and **@backstage/plugin-kubernetes-backend** dynamic plugins.
- You have configured the Kubernetes plugin to connect to the cluster using a **ServiceAccount**.
- The **ClusterRole** must be granted for custom resources (PipelineRuns and TaskRuns) to the **ServiceAccount** accessing the cluster.

NOTE

If you have the RHDH Kubernetes plugin configured, then the **ClusterRole** is already granted.

- To view the pod logs, you have granted permissions for **pods/log**.

- You can use the following code to grant the **ClusterRole** for custom resources and pod logs:

```
kubernetes:
  ...
  customResources:
    - group: 'tekton.dev'
      apiVersion: 'v1'
      plural: 'pipelineruns'
    - group: 'tekton.dev'
      apiVersion: 'v1'


...
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: backstage-read-only
rules:
  - apiGroups:
      - ""
    resources:
      - pods/log
    verbs:
      - get
      - list
      - watch
  ...
  - apiGroups:
      - tekton.dev
    resources:
      - pipelineruns
      - taskruns
    verbs:
      - get
      - list
```

  You can use the prepared manifest for a read-only **ClusterRole**, which provides access for both Kubernetes plugin and Tekton plugin.

- Add the following annotation to the entity's **catalog-info.yaml** file to identify whether an entity contains the Kubernetes resources:

```
annotations:
  ...

  backstage.io/kubernetes-id: <BACKSTAGE_ENTITY_NAME>
```

- You can also add the **backstage.io/kubernetes-namespace** annotation to identify the Kubernetes resources using the defined namespace.

```
annotations:
  ...

  backstage.io/kubernetes-namespace: <RESOURCE_NS>
```

- Add the following annotation to the **catalog-info.yaml** file of the entity to enable the Tekton related features in RHDH. The value of the annotation identifies the name of the RHDH entity:

```
annotations:
  ...

  janus-idp.io/tekton : <BACKSTAGE_ENTITY_NAME>
```

- Add a custom label selector, which RHDH uses to find the Kubernetes resources. The label selector takes precedence over the ID annotations.

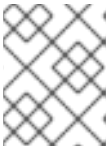
```
annotations:
  ...

  backstage.io/kubernetes-label-selector: 'app=my-app,component=front-end'
```

- Add the following label to the resources so that the Kubernetes plugin gets the Kubernetes resources from the requested entity:

```
labels:
  ...

  backstage.io/kubernetes-id: <BACKSTAGE_ENTITY_NAME>
```



NOTE

When you use the label selector, the mentioned labels must be present on the resource.

Procedure

- The Tekton plugin is pre-loaded in RHDH with basic configuration properties. To enable it, set the disabled property to false as follows:

```
global:
  dynamic:
    includes:
      - dynamic-plugins.default.yaml
    plugins:
      - package: ./dynamic-plugins/dist/janus-idp-backstage-plugin-tekton
        disabled: false
```

2.10.2. For users

2.10.2.1. Using the Tekton plugin in RHDH

You can use the Tekton front-end plugin to view **PipelineRun** resources.

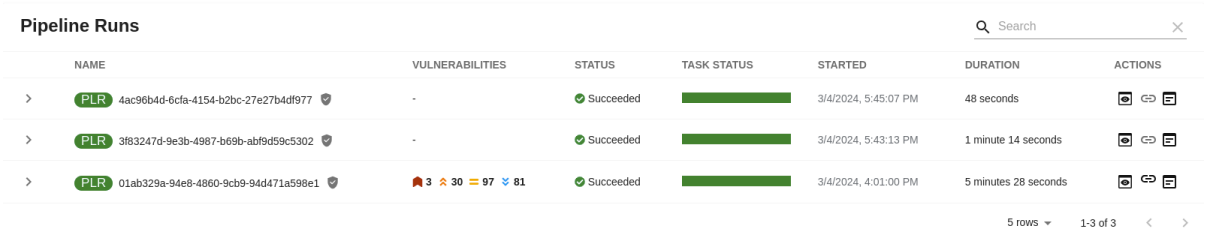
Prerequisites

- You have installed the Red Hat Developer Hub (RHDH).

- You have installed the Tekton plugin. For the installation process, see Installing and configuring the Tekton plugin.

Procedure

1. Open your RHDH application and select a component from the **Catalog** page.
2. Go to the **CI** tab.
   The **CI** tab displays the list of PipelineRun resources associated with a Kubernetes cluster. The list contains pipeline run details, such as **NAME**, **VULNERABILITIES**, **STATUS**, **TASK STATUS**, **STARTED**, and **DURATION**.



3. Click the expand row button besides PipelineRun name in the list to view the PipelineRun visualization. The pipeline run resource includes tasks to complete. When you hover the mouse pointer on a task card, you can view the steps to complete that particular task.