Open for Innovation
KNIME

KNIME Server Preview Functionality

KNIME AG, Zurich, Switzerland
Version 4.10 (last updated on 2021-11-26)

Table of Contents

General introduction to KNIME Server preview functionality

With the release of KNIME Server 4.10 we included two functionalities that are available as previews. That means that these functionalities are not always feature complete, or subject to change. The previews are provided to allow you to test the functionality and provide feedback that will help to shape the final product.

In case you have questions about any of the functionality in the previews please contact support@knime.com.

Embedded Queue-based Execution

KNIME Server 4.10 introduces an integrated message broker for communication between application server and executor, based on Apache Qpid (https://qpid.apache.org/). This can be thought of as using an architecture that corresponds closely to KNIME Server Distributed Executors, but runs on a single host, which is used by both application server and executor. One of the advantages of this setup is that application server and executor processes can be started separately, and can by run by separate users. This increases both resiliance and security.

The current plan is to switch KNIME Server to use the embedded queue as default execution mode with the next major release. This will effectively retire the current default, RMI-based execution.

Unlike distributed executors, it is not necessary to install a separate message broker such as RabbitMQ. Instead, Apache Qpid is bundled as part of all KNIME Server installations. At this point, it is not active by default, but needs a few easy steps to be enabled.

Doing so requires changes in two locations:

Application Server

Activate queue in `knime-server.config`. The file can be found in `<server-repository>/config/knime-server.config`. Set the parameter

`com.knime.enterprise.executor.embedded-broker=` to `true`

Server Executor

In the executor installation directory, add the following line to the `knime.ini` file, anywhere after the `-vmargs` line:

```
-Dcom.knime.enterprise.executor.msgq=amqp://knime:20knime16@localhost/
```

Unlike the RMI-based execution, using the Qpid message broker requires to startup the TomEE application server and the KNIME executor as two separate services.

For increased securiy, we recommend to run both services by different users.

For setting up the KNIME Server service, please follow the steps outlined in the KNIME Server Installation Guide. Installing the executor as a service follows the same procedure as described for Distributed Executors.

Once those two services are up and running, there will be no differences in terms of usage when compared to RMI-based execution.

OpenID Connect Authentication

With this preview feature, introduced with KNIME Server 4.10, authentication can be configured to use an OpenID Connect Identity Provider. This enables Single Single-On (SSO), using an existing Identity Provider.

When the feature is enabled and configured an authenticated user will be mapped to a user in the Tomcat realm. The user will be mapped using a configurable claim from the userinfo endpoint. For example if the claim *nickname* is chosen, let it be "john.doe", the user will be mapped to the internal user with the username "john.doe". This way the tomcat realms can still be configured according to the User Authentication section in the KNIME Server Administration Guide. Note that the password set for the user in the tomcat realm does not matter for the OAuth authentication. As long as the defined claim matches with a username in the database the user will be logged in with the assigned groups.

By default all authenticated users will be able to login in to the KNIME Server. Even if a user does not exist in the tomat realm, he will be able to access the KNIME Server. If access should be restricted, allowed login groups can be defined in the KNIME Server Configuration File, this way users that do not yet exist in the tomcat realm and are thus not assigned to any group cannot log in to the server.

Adding the following configuration option, will restrict the login to the specified groups:

`com.knime.server.login.allowed_groups =<group>,<group>,···`

Defines the groups that are allowed to log in to the server. Default value allows users from all groups.

Authentication Valve configuration for OpenID Connect (OAuth)

To enable OAuth authentication in the KNIME Server, the file <tomee-folder>/conf/Catalina/localhost/knime.xml has to be edited.

It is possible to configure the server such that both OAuth and basic authentication, using credentials, can be used at the same time. For the configuration the following parameters can be added to the authentication valve's definition:

`enableOAuth="<true|false>"`

Enables OAuth authentication. The default is false (OAuth authentication disabled).

`enableBasicAuthWithOAuth="<true|false>""`

Enables basic authentication along side OAuth if OAuth is enabled. The default is false (Basic authentication disabled when using OAuth).

From KNIME Server version 4.10.1 onward the "enableBasicAuth" option is renamed to "enableBasicAuthWithOAuth". For KNIME Server 4.10.0 "enableBasicAuth" must be used as configuration option. When updating from KNIME Server 4.10 to 4.10.1 the option must be renamed in the knime.xml file, only if the knime-tomcat.jar is replaced.

So the Valve entry should look similar to this:

```
<Valve className="com.knime.enterprise.tomcat.authenticator.KnimeServerAuthenticator"
enableSpnego="false" basicAuthPaths="/rest" formAuthPaths="/" secretKey="someSecreKey"
enableOAuth="true" enableBasicAuthWithOAuth="false"/>
```

The OpenID Connect Identity Provider Specific configuration is done by creating a *knime-oidc-config.json* file and placing it in <tomee-folder>/webapps/knime/WEB-INF/knime-oidc-config.json.

Here is an example of such a file, the parameters are explained below:

```
{
  "auth-server-url": "https://identitiy.provider/",
  "authorization-endpoint": "https://identitiy.provider/authorize",
  "token-endpoint": "https://identitiy.provider/token",
  "jwks-endpoint": "https://identitiy.provider/jwks",
  "userinfo-endpoint": "https://identitiy.provider/userinfo",
  "resource": "client-id",
  "credentials": {
    "secret" : "client-secret"
  },
  "additional-authorization-endpoint-parameters": "&additional-parameter=some-
value&some-other-parameter=some-value",
  "additional-scopes": "additional-scope another-scope",
  "principal-attribute": "claim-used-for-principal-mapping",
  "use-userinfo-for-principal": "true"
}
```

`additional-authorization-endpoint-parameters="&<additionalQueryParameters>"`

(OPTIONAL) Additional parameters used when calling the IdP's authorization endpoint.

`additional-scopes="<scope> <scope>"`

Space-separated list of additional scopes that should be requested when calling the IdP's authorization endpoint. The openid scope is always used when talking to the authorization endpoint.

`auth-server-url="<auth-server-url>"`

The base URL for the Identity Provider's OpenID Connect endpoint.

`principal-attribute="<claim>"`

The claim that should be used to map the authenticated users to the tomcat realm. By default this will use the *sub* claim. The *nickname* claim could be used for example, in this case, make sure that the corresponding scope is requested, when calling the authorization endpoint.

`use-userinfo-endpoint-for-principal="<true|false>"`

Set this to "true" if the Identity Provider only provides opaque access tokens, or access tokens that do not contain the necessary claims to map the authorized user in the tomcat realm. This will probably be set to true in most cases.

KNIME Server Client Configuration for the KNIME Analytics Platform

In order to enable OAuth authentication for clients of a KNIME Server configured for OpenID Connect, some additional configuration has to be added to the server-repository>/config/knime-server.config file.

The following show what the additional configuration could look like, the configuration parameters are described below:

```
com.knime.server.authentication.oauth.authorization_endpoint=https://identitiy.provider/
authorize
com.knime.server.authentication.oauth.token_endpoint=https://identitiy.provider/token
com.knime.server.authentication.oauth.client_id=client-id
com.knime.server.authentication.oauth.client_secret=client-secret
com.knime.server.authentication.oauth.redirect_ports=8888,8881,8882
com.knime.server.authentication.oauth.scope=additional-scope, another-scope
com.knime.server.authentication.types=OAuth,Credentials
com.knime.server.authentication.oauth.additional_query_params=&additional-
parameter=some-value&some-other-parameter=some-value
com.knime.server.authentication.oauth.token_refresh_rate=5m
```

`com.knime.server.authentication.oauth.authorization_endpoint=<authorization-endpoint>`

The Identitiy Provider's authorization endpoint.

`com.knime.server.authentication.oauth.redirect_ports=<port>,<port>,<port>`

(OPTIONAL) A comma separated list of ports that should be used for the authorization redirect. If no list is given a random port will be chosen for each authorization. Some Identity Providers might not support wildcards, in this case a list of configured ports should be provided here.

`com.knime.server.authentication.oauth.scope=<scope>,<scope>,<scope>`

Comma-separated list of additional scopes that should be requested when calling the Identity Providers's authorization endpoint. The scopes should match the scopes in the knime-oidc-config.json on the server side. The **openid** scope should be provided in any case, along with the additional scopes also defined in the knime-oidc-config.json.

`com.knime.server.authentication.types=<type>,<type>`

Comma-separated list of authentication types. Possible values are "OAuth" and "Credentials". This should match the authentication valve configuration in <tomee-folder>/conf/Catalina/localhost/knime.xml.

`com.knime.server.authentication.oauth.additional_query_params=<query-string>`

Additional parameters that should be used when calling the Identity Provider's authorization endpoint, provided as a query string.

`com.knime.server.authentication.oauth.token_refresh_rate=<duration with unit, e.g. 5m, 30m or 2h>`

(OPTIONAL) If the access token is opaque, the expiry date of the access token cannot be determined, so a refresh rate can be introduced, to refresh the token.

Known Issues

- (Webportal) When restricting login groups and a user is not part of a login-allowed group he will be redirected to a non-functional login-page.
- (Webportal) When using OAuth, the logout is not functional and might lead to an unexpected error on the Identity Provider side. To end the session and login as a different user, cookies must be cleared.
- (Analytics Platform) Being logged-in while restarting the Server leads to unexpected errors.

`com.knime.server.authentication.oauth.token_endpoint=<token-endpoint>`

The Identitiy Provider's token endpoint.

`com.knime.server.authentication.oauth.client_id=<client-id>`

The application's client ID.

KNIME AG
Technoparkstrasse 1
8005 Zurich, Switzerland
www.knime.com
info@knime.com