Open for Innovation
KNIME

KNIME Server Administration Guide

KNIME AG, Zurich, Switzerland
Version 4.10 (last updated on 2021-12-10)

Table of Contents

Introduction

This guide covers in detail the configuration options for KNIME Server.

If you are looking to install KNIME Server, you should first consult the KNIME Server Installation Guide.

For guides on connecting to KNIME Server from KNIME Analytics Platform, or using KNIME WebPortal please refer to the following guides:

- KNIME Explorer User Guide
- KNIME WebPortal User Guide

There are additional resources such as the KNIME Server Advanced Setup Guide and KNIME Server Preview Functionality Guide.

Release Notes

KNIME Server 4.10 is a feature release of the 4.x release line. All clients that have worked with KNIME Server 4.9 will continue to work with KNIME Server 4.10 without restrictions.

To find out which version of KNIME Server you are currently running, you can check the Administration pages on the WebPortal.

New Features

For a list that includes the new Analytics Platform 4.1 features see here.

Highlighted new functionality is:

- OAuth authentication (preview)
- Call workflow action (what's new)
- Configuration dialogs (what's new)
- Execution via embedded message broker
- Improvements to Server Managed Customizations

A detailed changelog for KNIME Server 4.10 is also available.

Configuration options

Since KNIME Server 4.10 it is possible to configure workflows before execution. Starting with KNIME Server 4.10.1 these configurations are encrypted using the secret key defined in

```
<tomee-folder>/conf/Catalina/localhost/knime.xml
```

While the behavior remains the same as for KNIME Server 4.10 and is backward compatible, downgrading from KNIME Server 4.10.1 to a previous version will result in losing job and scheduled job information and thus is not recommended. This information will also be lost in case the secret key changes.

New options in knime-server.config

The following new options are available in the knime-server.config. Full details can be found at KNIME Server configuration file options.

- `com.knime.enterprise.executor.embedded-broker=<true|false>`

Changes to default executor rotation

In previous releases, the KNIME Server executor was rotated once a day by default. This means that after 24 hours, an executor would not accept any new jobs, after which a new executor starts up and takes its place. After a transitional period, the old executor was retired.

This overlap during the transitional period comes with a few undesired side effects, most notably a lack of efficiency when it comes to resource usage. For this reason, we decided to deactivate this function by default. By doing so, there is no need to factor in an additional executor when allocating available memory.

Note that we did not remove the rotation functionality, but instead only deactivated it by default. If needed, you can still activate it by adjusting the parameter `com.knime.server.executor.max_lifetime` in knime-server.config.

Server Managed Customizations

Controlling the available update sites via Server Managed Customizations has been extended. It’s now possible to add multiple update sites and also disable the default update sites.

The options for adding a single update site have been deprecated and will be removed in the future. You can easily adjust your templates by simply adding an "s" to the property names (`updateSite.uri` to `updateSite.uris` and `updateSite.name` to `updateSite.names`). See Client customization for details.

Controlling the available update sites via Server Managed Customizations has been extended. It’s now possible to add multiple update sites and also disable the default update sites.

The options for adding a single update site have been deprecated and will be removed in the future. You can easily adjust your templates by simply adding an "s" to the property names (`updateSite.uri` to `updateSite.uris` and `updateSite.name` to `updateSite.names`). See Client customization for details.

Local file system access by KNIME workflows

Several KNIME nodes have been revised to use a new shared framework for file access (see below for a list of nodes). When executing on KNIME Server, a new preference controls whether those nodes can access the local file system of the KNIME Server Executor or not. Currently local file system access is allowed by default, however the default will change with the next release (KNIME Server 4.11).

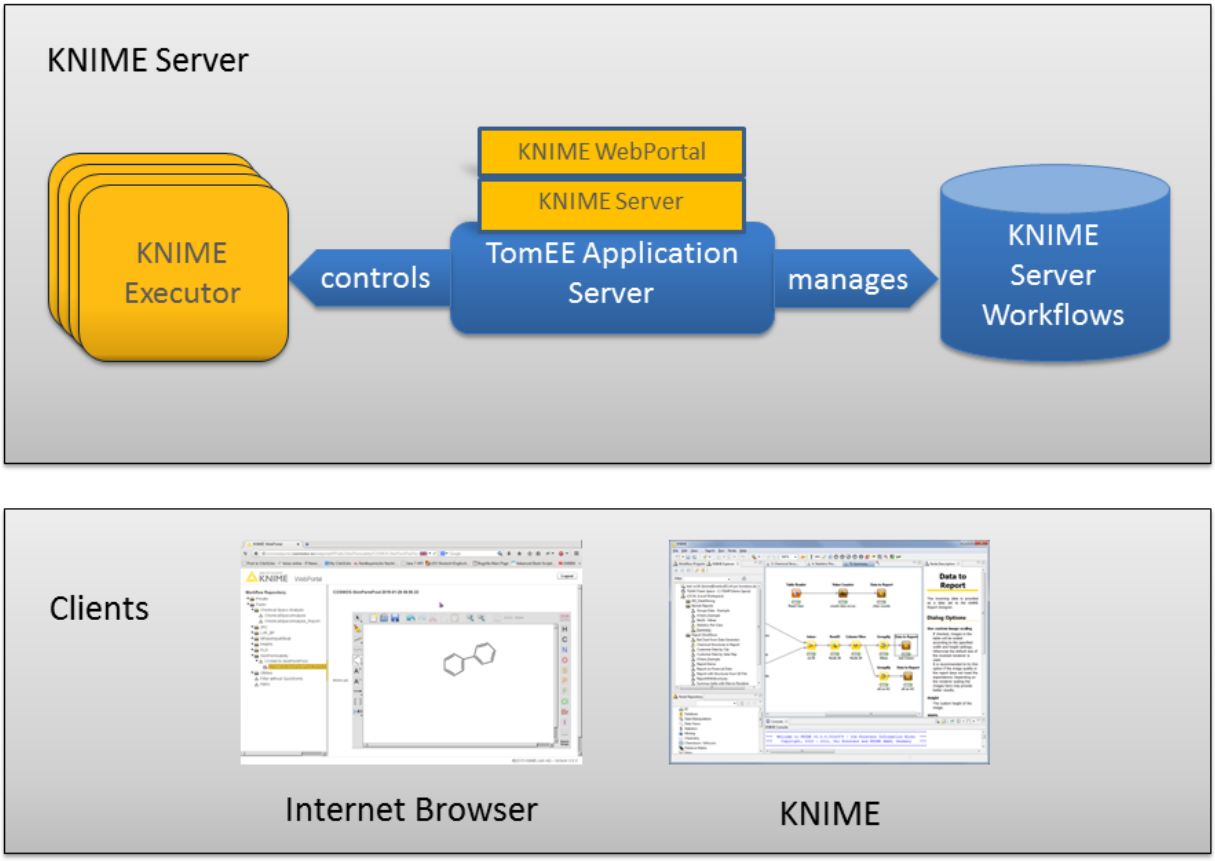
To disallow local file system access, add the following line to the `preferences.epf` or customization profile used by your KNIME Server Executor(s):

```
/instance/org.knime.filehandling.core/allow_local_fs_access_on_server=false
```

This preference currently only affects the following KNIME nodes: Excel Reader (XLS), Excel Writer (XLS), Excel Sheet Appender (XLS), Line Reader. More nodes will follow over the next releases.

Server architecture

KNIME Server is a Java Enterprise Application, and the KNIME WebPortal a standard Java Web Application, both installed on a TomEE application server. TomEE is an extended Tomcat server, the blue box in the middle of the figure below. Users can log in to the server and the server will authenticate against any authentication source provided by Tomcat.



One of the main tasks of KNIME Server is to manage and control the server's repository. Workflows uploaded to the server go through the server application and are stored in the repository which is just a folder on the server's file system (the blue cylinder on the right in the diagram). Access to the stored workflows is controlled in KNIME Server and access rights for the workflows can be manipulated from KNIME Explorer once the client side server extensions are installed.

Workflow execution on the server is carried out by a KNIME Executor. The KNIME Executor is a persistent headless instance of a normal KNIME Analytics Platform application (leftmost element in the diagram above). The server can, depending on the installation, use either one executor for all workflow executions, or a separate executor instance for each authenticated user.

It is important to note that workflows can only be successfully loaded and executed on the server, if the executor has the required features installed and is of the same version (or newer) than the KNIME Analytics Platform version that was used to create the workflow.

Server configuration files and options

KNIME Server configuration file

KNIME Server is configured by a knime-specific configuration file named `knime-server.config`. The file can be found in `<server-repository>/config/knime-server.config`. Most of the parameters defined in this file can be changed at runtime and will take effect as soon as possible. Default values will be used for empty or missing configuration options.

The section KNIME Server configuration file options contains a comprehensive list of all configuration options and explanations.

KNIME Server configuration file options

Below you will find a table with all supported configuration options (in alphabetical order). Some of them are described in more detail in later sections. The options can be set in the file `<server-repository>/config/knime-server.config`.

For Windows users: For paths in the server configuration file either use forward slashes ("/") or double backslashes ("\\"). A single backslash is used to escape characters.

The following annotations to the table, provide some additional information about which executor type is affected, and whether changes take effect at runtime, or require a server restart.

[ST]	changes take effect after a restart of KNIME Server
[RT]	changes can take effect at runtime
[RE]	changes only affect RMI executors
[DE]	changes only affect distributed executors (see here.)

Some options can be set as property in the `knime-server.config` file as well as by defining an environment variable (Env). The environment variable changes will only take effect after a restart of KNIME Server. If the environment variable for an option is set, the property in the configuration file will be ignored.

`com.knime.server.admin_email=<email>,<email>,⋯` **[RT]**

A comma separated list of email addresses that will get notified when there is a problem with the server, e.g. the license is about to expire or the maximum number of users has been reached.

`com.knime.server.canonical-address=<URL to server>` **[RT]**

The communication between executor and server is performed through the server's REST interface. In case auto-detection of the server's address doesn't work correctly, you have to specify the canonical address here, e.g. http://knime-server:8080/. This option is not required if server and executor are running on the same computer. See also section enabling workflow execution below for more details.

Env: `KNIME_SERVER_CANONICAL_ADDRESS=<URL to server>`

`com.knime.server.config.watch=<true|false>` **[ST]**

If set to true changes to the configuration file are applied immediately without a server restart. Default is `false`, i.e. all changes will require a server restart.

`com.knime.server.csp-report-only=<true|false>` **[RT]**

Tells the browser to still serve content that violates the Content-Security-Policy and instead display a warning. By setting the Content-Security-Policy-Report-Only header rather than the Content-Security-Policy header (defaults to false).

`com.knime.server.default_mount_id=<mount ID>` **[RT]**

Specifies the name of the default mount ID. This is fetched, when clients set up their mount point to the server. Defaults to the server's hostname.

Env: `KNIME_SERVER_DEFAULT_MOUNT_ID=<mount ID>`

`com.knime.enterprise.executor.msgq=amqp://<user>:<password>@<rabbitmq-host>/<vhost>` **[DE][ST]**

URL to the RabbitMQ virtual host. In case RabbitMQ High Available Queues are used, simply add additional `<rabbitmq-host>:<port>` separated by commas to the initial `amqp` address: `com.knime.enterprise.executor.msgq=amqp://<username>:<password>@rabbitmq-host/knime-server,amqp://<rabbitmq-host2>:<port2>,amqp://<rabbitmq-host3>:<port3>` Note, this is supported with KNIME Server 4.10.5 and 4.11.3 onward.

Env: `KNIME_EXECUTOR_MSGQ=amqp://<user>:<password>@<rabbitmq-host>/<vhost>`

`com.knime.enterprise.executor.msgq.connection_retries=<value>` **[DE][ST]**

Defines the maximum number of connection retries for the message queue, that should be performed during server startup. The delay between retries is 10 seconds. The default is `5`, `<value>` has to be an integer value greater or equal to 1.

Env: `KNIME_MSGQ_CONNECTION_RETRIES=<value>`

`com.knime.server.executor.blacklisted_nodes=<node>,<node>,…` **[RT]**

Specifies nodes that are blacklisted by the server, i.e. which aren't allowed to be executed. For blacklisting a node you have to provide its factory name. Wildcards (*) are supported. For more information see here.

`com.knime.server.executor.knime_exe=<path to knime executable>` **[RE][RT]**

Specifies the KNIME executable that is used to execute flows on the server. Default is none (no execution available on the server).

`com.knime.server.executor.max_instances=<number>`**[RE][RT]**

Specifies the maximum number of KNIME Executors used on the server (defaults to `10`), if multiple KNIME Executors are used.

`com.knime.server.executor.max_lifetime=<duration with unit, e.g. 60m, 36h, or 2d>` **[RE][RT]**

Specifies the time in minutes after which an executor is retired and a new instance is created. By default, this function is turned off (defaults to `-1`). Positive numbers enable executor rotation. E.g., setting this parameter to 1d will rotate executors once a day.

`com.knime.server.executor.prestart=<true|false>` **[RE][ST]**

Specifies whether an executor should be started during server startup or if it should be started on-demand when the first workflow is being executed. Default is to prestart the executor. This setting has no effect if per-user executors are used (i.e. if `com.knime.server.executor.sudo_cmd` is defined). In this case executors are always started on demand.

`com.knime.server.executor.reject_future_workflows=<true|false>` **[RT]**

Specifies whether the executor should reject loading workflows that have been create with future versions. For new installations the value is set to true. If no value is specified the executor will always try to load and execute any workflow by default.

`com.knime.server.executor.skip_teamspace_mount=<true|false>` **[RE][RT]**

Specifies whether mounting the server's workflow repository in the KNIME Executor (as a TeamSpace) should be skipped. Default is to mount the workflow repository.

`com.knime.server.executor.start_port=<port>` **[RE][ST]**

Specifies the start port that the server uses to communicate with the KNIME Executor. Default is `50100`. With multiple executors and/or automatic executor renewal multiple consecutive ports are used.

`com.knime.server.executor.sudo_cmd=<path to sudo command>` **[RE][RT]**

Specifies the `sudo` command. Default is to not use `sudo` i.e. all user share the same RMI instance.

`com.knime.server.executor.update_metanodelinks_on_load=<true|false>` **[RT]**

Specifies whether component links in workflows should be updated right after the workflow has been loaded in the KNIME Executor. Default is not to update component links.

`com.knime.server.job.default_load_timeout=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Specifies how long to wait for a job to get loaded by an executor. If the job does not get loaded within the timeout, the operation is canceled. The default is `1m`. This timeout is only applied if no explicit timeout has been passed with the call.

`com.knime.server.job.default_report_timeout=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Specifies how long to wait for a report to be created by an executor. If the report is not created within the timeout, the operation is canceled. The default is `1m`. This timeout is only applied if no explicit timeout has been passed with the call.

`com.knime.server.job.default_swap_timeout=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Specifies how long to wait for a job to be swapped to disk. If the job is not swapped within the timeout, the operation is canceled. The default is `1m`. This timeout is only applied if no explicit timeout has been passed with the call (e.g. during server shutdown).

`com.knime.server.job.discard_after_timeout=<true|false>` **[RT]**

Specifies whether jobs that exceeded the maximum execution time should be canceled and discarded (`true`) or only canceled (`false`). May be used in conjunction with `com.knime.server.job.max_execution_time` option. The default (`true`) is to discard those jobs.

`com.knime.server.job.max_execution_time=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Allows to set a maximum execution time for jobs. If a job is executing longer than this value it will be canceled and eventually discarded (see `com.knime.server.job.discard_after_timeout` option). The default is unlimited job execution time. Note that for this setting to work, `com.knime.server.job.swap_check_interval` needs to be set a value **lower** than `com.knime.server.job.max_execution_time`.

`com.knime.server.job.max_lifetime=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Specifies the time of inactivity, before a job gets discarded (defaults to `7d`), negative numbers disable forced auto-discard.

`com.knime.server.job.max_time_in_memory=<duration with unit, e.g. 60m, 36h, or 2d>` **[RT]**

Specifies the time of inactivity before a job gets swapped out from the executor (defaults to `60m`), negative numbers disable swapping.

`com.knime.server.job.status_update_interval=<duration with unit, e.g. 500ms, 2s, or 5m>` **[RE][RT]**

Specifies the interval at which the running executor instances are checked for unnoticed status changes and if they are still alive. Default is every `60s`.

`com.knime.server.job.swap_check_interval=<duration with unit, e.g. 30s, 1m, or 1h>` **[RT]**

Specifies the interval at which the server will check for inactive jobs that can be swapped to disk. Default is every `1m`.

`com.knime.server.login.allowed_groups =<group>,<group>,⋯` **[RT]**

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

Env: `KNIME_LOGIN_ALLOWED_GROUPS=<group>,<group>,⋯`

`com.knime.server.login.consumer.allowed_accounts =<account>,<account>,⋯` **[RT]**

Defines account names that are allowed to log in to the server as *consumer*. Default value allows login as consumer for all users.

Env: `KNIME_CONSUMER_ALLOWED_ACCOUNTS=<account>,<account>,⋯`

`com.knime.server.login.consumer.allowed_groups =<group>,<group>,⋯` **[RT]**

Defines the groups that are allowed to log in to the server as *consumer*. Default value allows login as consumer from all groups.

Env: `KNIME_CONSUMER_ALLOWED_GROUPS=<group>,<group>,⋯`

`com.knime.server.login.jwt-lifetime=<duration with unit, e.g. 12h or 30d>` **[RT]**

Defines the maximum lifetime of JSON Web Tokens issued by the server. The default value is `30d`. A negative value allows unrestricted tokens (use this value with care because there is no way to revoke issued tokens).

`com.knime.server.login.user.allowed_accounts =<account>,<account>,⋯` **[RT]**

Defines account names that are allowed to log in to the server as *user*. Default value allows login as user for all users.

`com.knime.server.login.user.allowed_groups =<group>,<group>,⋯` **[RT]**

Defines the groups that are allowed to log in to the server as a *user*. Default value allows login as user from all groups.

`com.knime.server.report_formats=<formats>` **[RT]**

Defines the different formats available for report generation as a comma separated list of values. Possible values are `html`, `pdf`, `doc`, `docx`, `xls`, `xlsx`, `ppt`, `pptx`, `ps`, `odp`, `odt` and `ods`. If this value is empty or not set the default list of formats is `html`, `pdf`, `docx`, `xlsx` and `pptx`.

`com.knime.server.repository.update_recommendations_at=<time>` **[RT]**

Defines a time during the day (in ISO format, i.e. 24h notation, e.g. `21:15`) at which the node recommendations for the workflow coach are updated based on the current workflow repository contents. Default is undefined which means that no node recommendations will be computed and provided by the server.

`com.knime.server.server_admin_groups=<group>,<group>,⋯` **[RT]**

Specifies the admin group(s). Users belonging to at least one of these groups are considered KNIME Server admins (not TomEE server admins). Default is no admin groups.

Env: `KNIME_SERVER_ADMIN_GROUPS=<group>,<group>,⋯`

`com.knime.server.server_admin_users=<user1>,<user2>,⋯` **[RT]**

Specifies the user(s) that are KNIME Server admins (not TomEE admins). Default is no users.

`com.knime.server.user_directories.directory_location=<location>` **[ST]**

Specifies the base directory in which user directories shall be created on first login. When the base directory is created its `<owner>` is set to the one defined with `com.knime.server.user_directories.parent_directory_owner`. Also all non existing directories under `<location>` will be created and their owner set to `<owner>`. The permissions of the created directories are: `owner: rwx, world: r--`. If left empty no user directories will be created and all `com.knime.server.user_directories` options will be ignored. Note that only logins via the KNIME Analytics Platform will cause a user directory to be created.

`com.knime.server.user_directories.parent_directory_owner=<owner>` **[ST]**

Specifies the owner of the base directory created at `<location>` (see `com.knime.server.user_directories.directory_location`). If left empty the default value `knimeadmin` will be used.

`com.knime.server.user_directories.owner_permissions=<permission>` **[ST]**

Specifies the permissions of the owners (users themselves) for their created user directories. The defined permissions have to be in a block of 3 characters (r,w,x,-), e.g. `rwx` or `r-x`. If left empty the default value `rwx` is used.

`com.knime.server.user_directories.inherit_permissions=<true|false>` **[ST]**

Specifies if the permissions of the created user directories shall be inherited from their parent directory. If left empty the default value `false` is used.

`com.knime.server.user_directories.groups=<group1>:<permission1>,<group2>:<permission2>,⋯` **[ST]**

Specifies the permissions of groups for the created user directories. The defined permissions have to be in a block of 3 characters (r,w,x,-), e.g. `rwx` or `r-x`. If left empty no group permissions are set.

`com.knime.server.user_directories.users=<user1>:<permission1>,<user2>:<permission2>,⋯` **[ST]**

Specifies the permissions of users for the created user directories. The defined permissions have to be in a block of 3 characters (r,w,x,-), e.g. `rwx` or `r-x`. If left empty no user permissions are set.

`com.knime.server.user_directories.world_permissions=<permission>` **[ST]**

Specifies the permissions of others for the created user directories. The defined permissions have to be in a block of 3 characters (r,w,x,-), e.g. `rwx` or `r-x`. If left empty the default value `r--` is used.

`com.knime.server.webportal.csp=<CSP statement>` **[RT]**

Specified a custom Content Security Policy for the WebPortal. It may be necessary to override the default if you are using custom JavaScript views that load external resources. The default works for all standard KNIME views.

`com.knime.server.webportal.debug=<true|false>` **[RT]**

Enables or disables a debug mode for the WebPortal. In debug mode JavaScript and CSS sources are included in their non-minified version and log messages might be printed to the console of the browser.

`com.knime.server.webportal.disable_report_preview=<true|false>` **[RT]**

Disables the report preview in the KNIME WebPortal. Default is to show report previews.

`com.knime.server.webportal.disable_warning_messages=<true|false>` **[RT]**

Disables warnings messages at the end of a workflow execution on the KNIME WebPortal. Default is to show warnings messages.

`com.knime.server.webportal.hide_version=<true|false>` **[RT]**

Hides the server's version in the KNIME WebPortal. Default is to show the version number.

`com.knime.server.webportal.ie_compatibility=<IE version identifier>` **[RT]**

This option allows you to set the IE compatibility mode that the KNIME WebPortal sends to the browser. Default is not to send any compatibility information.

`com.knime.server.webportal.restrict_x_frame_options=<value>` **[RT]**

Sets the value of the HTTP-header `X-Frame-Options`. `<value>` must be one of `DENY`, `SAMEORIGIN` or `ALLOW-FROM xxx`, where `xxx` needs to be replaced with the URL of the embedding page. If this option is not present in the configuration file, the HTTP-header `X-Frame-Options` is not sent. See also avoiding clickjacking attacks.

`com.knime.server.webportal.sketcher_page=<relative URL of Sketcher Page>` **[RT]**

Define the location of the main sketcher html-document. Helpful when the sketcher is deployed as static resources under a different context root. Note that the KNIME WebPortal and the sketcher need to be in the same domain. Otherwise cross-domain scripting would occur, which is considered a security threat in all major browsers and thus not allowed.

`com.knime.server.webportal.sketcher_size=<width x height, e.g. 300x300>` **[RT]**

Define the size of the sketcher iframe. `300x300` is the default value for the Marvin Sketcher.

`com.knime.server.webportal.title_label=<Server Name>` **[RT]**

Define an additional title label displayed to the right of the KNIME WebPortal logo.

In the client KNIME Analytics Platform, these options are supported by the KNIME Server: Add them to the `knime.ini` file. After the `-vmargs` line, each in a separate line.

`-Dcom.knime.server.server_address=<KNIME server>`

Sets the `<KNIME server>` as the default Workflow Server in the client view.

Default mount ID

KNIME supports mountpoint relative URLs using the `knime` protocol (see the KNIME Explorer User Guide for more details). Using this feature with KNIME Server requires both the workflow author and their collaborator to use the shared Mount IDs. With this in mind, you can now set a common name (Mount ID) for the server to all users.

The default name for your server can be specified in the configuration file:

```
com.knime.server.default_mount_id=<server name>
```

Blacklisting nodes

You might want to prevent the usage of certain nodes on the executor of KNIME Server. While you can decide, which extensions you install for the executor there might be nodes in the basic installation of KNIME Analytics Platform or in a required extension that shouldn't be used.

The configuration option

```
com.knime.server.executor.blacklisted_nodes=<node>,<node>,...
```

allows you to define a list of nodes that should be blocked by the executor. This list also supports wildcards (*). If a workflow contains a blacklisted node the executor will throw an error and abort loading the workflow.

To blacklist a node you have to provide the full name of the node factory. The easiest way to determine the factory names of the nodes you want to block is to create a workflow with all nodes that should be blacklisted. After saving the workflow you are able to access the `settings.xml` of each node under `<knime-workspace>/<workflow>/<node>/settings.xml`. The factory name can be found in the entry with key `"factory"`.

The following shows an example on how to block the Java Snippet nodes. The factory information for the Java Snippet node is

```
<entry key="factory" type="xstring"
value="org.knime.base.node.jsnippet.JavaSnippetNodeFactory"/>
```

To block the Java Snippet node we simply provide the value (without the quotes)

```
com.knime.server.executor.blacklisted_nodes=org.knime.base.node.jsnippet.JavaSnippetNode
Factory
```

The factory names for Java Snippet (simple), Java Snippet Row Splitter, and Java Snippet Row Filter are

```
org.knime.ext.sun.nodes.script.JavaScriptingNodeFactory
org.knime.ext.sun.nodes.script.node.rowsplitter.JavaRowSplitterNodeFactory
org.knime.ext.sun.nodes.script.node.rowfilter.JavaRowFilterNodeFactory
```

Since they all share the same prefix, we append n factory name making use of wildcards:

```
com.knime.server.executor.blacklisted_nodes=org.knime.base.node.jsnippet.JavaSnippetNode
Factory,org.knime.ext.sun.nodes.script.*Java*
```

While users are still able to upload workflows containing these nodes, the executor won't load a workflow containing any of them.

KNIME Executor job handling

Job swapping

Jobs that are inactive for a period of time may be swapped to disc and removed from the executor to free memory or executor instances. A job is inactive if it is either fully executed or waiting for user input (on the KNIME WebPortal). If needed, it will be retrieved from disk automatically.

The configuration option

```
com.knime.server.job.max_time_in_memory=<duration with unit, e.g. 60m, 36h, or 2d>
```

controls the period of inactivity allowed before a job will be swapped to disk (default = `60m`). If you specify a negative number this feature is disabled and inactive jobs stay in memory until they are discarded.

Note: There are certain flows that will not be restored in the exact same state that it was in, before it got swapped out. For example, if a flow gets swapped with a loop partially executed, this loop iteration will be reset and the loop execution is restarted.

Job auto-discard

There is an additional threshold for inactivity of a job after which it may be discarded automatically. A discarded job due to inactivity cannot be recovered. The time threshold for a job to be automatically discarded is controlled by setting

```
com.knime.server.job.max_lifetime=<duration with unit, e.g. 60m, 36h, or 2d>
```

The default value (if the option is not set) is `7d`.

Restarting the executor

KNIME Server will periodically recycle its workflow executor. This process should not have any effect on workflow execution and should not significantly impact end-users. The server starts a second instance of the executor and loads future workflow jobs using the new instance — retiring the old executor instance after all jobs in the retired executor are finished.

```
com.knime.server.executor.max_lifetime=<duration with unit, e.g. 60m, 36h, or 2d>
```

Controls the maximum lifetime of an executor, after which it will be recycled (default = `-1`, a negative value will disable this feature)

Note: If this feature is enabled, the server must have enough resources to host two KNIME Executors (or more, in the case of multiple executors per user). Each executor instance requires at least the amount of memory specified in the `knime.ini` file.

Preferences file

If the KNIME Executor requires certain preferences (e.g. database drivers), you need to provide a preference file to the server that is read by every started KNIME Executor.

1. Start KNIME (with an arbitrary workspace).
2. Set all preferences via "File" → "Preferences") and export the preferences via "File" → "Export Preferences". This step can also be performed on a client computer but make sure that any paths you set in the preferences are also valid on the server.
3. Copy the exported preferences file as `preferences.epf` into `<server-repository>/config`.

Note: Make sure to specify the paths of all database drivers in the new preference page, in order to be able to execute workflows with database nodes. The page is available in the "KNIME" → "Database Drivers" category of the preferences.

Adding JDBC drivers to executor (headless executor)

In order to be able to execute workflows that contain database nodes that use custom or proprietary JDBC driver files on KNIME Server, the `preferences.epf` file must contain the path to the JDBC jar file, or the folder containing the JDBC driver. This may be specified in the KNIME Analytics Platform (executor) GUI and the `preferences.epf` file exported as described in the above section. This is the recommended route for systems that have graphical access to the KNIME Analytics Platform (executor).

Some systems do not have graphical access to the KNIME Analytics Platform (executor) GUI. In that case the `preferences.epf` can be manually created, or created on an external machine and copied into location on the server. The relevant lines that must be contained in the `preferences.epf` file are:

```
file_export_version=3.0
\!/=
/instance/org.knime.workbench.core/database_drivers=/path/to/driver.jar;/path/to/driver-
folder
/instance/org.knime.workbench.core/database_timeout=60
```

Note that `driver.jar` may also reference a folder in some cases (e.g. MS SQL Server and Simba Hive drivers).

We've bundled a file called `preferences.epf.template` into the `<server-repository>/config` folder. In order for those preferences to be used, you must edit the file as appropriate, and

move it so that it is named `preferences.epf`.

If you are using distributed executors, please see the Server-managed Customization Profiles section of the KNIME Database Extension Guide for how to distribute JDBC drivers.

knime.ini file

You might want to tweak certain settings of this KNIME instance, e.g. the amount of available memory or set system properties that are required by some extensions. This can be changed directly in the `knime.ini` in the KNIME executor installation folder.

KNIME Server will read the `knime.ini` file next to the KNIME executable and create a custom ini file for every executor that is started. However, if you use a shell script that prepares an environment the server may not be able to find the ini file if this start script is in a different folder. In this case the `knime.ini` file must be copied to `<server-repository>/config/knime.ini`. If this file exists, the server will read it instead of searching for a `knime.ini` next to the executable or start script.

Log files

There are several log files that could be inspected in case of unexpected behavior:

TomEE server log

Location: `<tomee-folder>/logs/catalina.yyyy-mm-dd.log`
This file contains all general TomEE server messages, such as startup and shutdown. If TomEE does not start or the KNIME Server application cannot be deployed, you should first look into this file.

Location: `<tomee-folder>/logs/localhost.yyyy-mm-dd.log`
This file contains all messages related to the KNIME Server operation. It does not include messages from the KNIME Executor!

For new installations these files are kept for 90 days before being removed. The default behaviour can be changed by editing the `<tomee-folder>/conf/logging.properties` file and amending any entries with:

```
1catalina.org.apache.juli.FileHandler.maxDays = 90
```

KNIME executor log

Location: `<server-repository>/runtime/runtime_knime-rmi_<suffix>/.metadata/knime/knime.log`
Depending on the configuration, the suffix is either a number or a username, or a combination of both.
This file contains messages from the KNIME Executor that is used to execute workflows on the server (for manually triggered execution, scheduled jobs, and also for generated reports, if KNIME Report Server is installed).
Also useful in some cases is the Eclipse log file `<server-repository>/runtime/runtime_knime—rmi_<suffix>/.metadata/.log`

KNIME Analytics Platform (client) log

Location: `<local workspace>/.metadata/knime/knime.log`
This file contains messages of the client KNIME application. Messages occurring during server communications are logged there. The Eclipse log of this application is in `<local workspace>/.metadata/.log`

Email notification

KNIME Server allows users to be notified by email when a workflow finishes executing. The emails are sent from a single email address which can be configured as part of the web application's mail configuration. If you don't want to enable the email notification feature, no email account is required. You can always change the configuration and enter the account details later.

Setting up the server's email resource

The email configuration is defined in the web application context configuration file which is `<tomee-folder>/conf/Catalina/localhost/knime.xml` (or `com.knime.enterprise.server.xml` or similar). The installer has already created this file. In order to change the email configuration, you have to modify or add attributes of/to the `<Resource name="mail/knime" ⋯ />` element. All configuration settings must be added as attributes to this element. The table below shows the list of supported parameters (see also the JavaMail API documentation). Note that the mail resource's name must be `mail/knime` and cannot be changed.

Name	Value
`mail.smtp.from`	Address from which all mails are sent
`mail.smtp.host`	SMTP server, required
`mail.smtp.port`	SMTP port, default `25`
`mail.smtp.auth`	Set to `true` if the mail server requires authentication; optional
`mail.smtp.user`	Username for SMTP authentication; optional
`password`	Password for SMTP authentication; optional
`mail.smtp.starttls.enable`	If `true`, enables the use of the STARTTLS command (if supported by the server) to switch the connection to a TLS-protected connection before issuing any login commands. Defaults to false.

Name	Value
`mail.smtp.ssl.enable`	If set to true, use SSL to connect and use the SSL port by default. Defaults to `false`.

If you do not intend to use the email notification service (available in the KNIME WebPortal for finished workflow jobs), you can skip this step.

Note that the mail configuration file contains the password in plain text. Therefore, you should make sure that the file has restrictive permissions.

User authentication

As described briefly in the Server architecture section it is possible to use any of the authentication methods available to Tomcat in order to manage user authentication. By default the KNIME Server installer configures a database (H2) based authentication method. Using this method it is possible for admin users to add/remove users/groups via the AdminPortal using a web-browser. Other users may change their password using this technique.

For enterprise applications, use of LDAP authentication is recommended, and user/group management is handled in Active Directory/LDAP itself.

In all cases the relevant configuration information is contained in the

```
`<Realm className="org.apache.catalina.realm.LockOutRealm">`
```

tag in `<tomee-folder>/conf/server.xml`.

The default configuration uses a `CombinedRealm` which allows multiple authentication methods to be used together. Examples for each of database, file and LDAP authentication are contained within the default installation. Configuration of all three authentication methods are described briefly in the following sections. In all cases the Tomcat documentation should be considered the authoritative information source.

LDAP authentication

LDAP authentication is the recommended authentication in any case where an LDAP server is available. If you are familiar with your LDAP configuration you can add the details during installation time, or edit the `server.xml` file post installation. If you are unfamiliar with your LDAP settings, you may need to contact your LDAP administrator, or use the configuration details for any other Tomcat based system in your organization. Please refer to the KNIME Server Advanced Setup Guide for details on setting up LDAP.

Connecting to an SSL secured LDAP server

In case you are using encrypted LDAP authentication and your LDAP server is using a self-signed certificate, Tomcat will refuse it. In this case you need to add the LDAP server's certificate to the global Java keystore, which is located in `<jre-folder>/lib/security/cacerts`:

```
keytool -import -v -noprompt -trustcacerts -file \
      <server certificate> -keystore <jre>/lib/security/cacerts \
      -storepass changeit
```

Alternatively you can copy the `cacerts` file, add your server certificate, and add the following two system properties to `<tomee-folder>/conf/catalina.properties`:

```
javax.net.ssl.trustStore=<copied keystore>
javax.net.ssl.keyStorePassword=changeit
```

Single-sign-on with LDAP and Kerberos

It is possible to use Kerberos in combination with LDAP for Single-Sign-On for authentication with KNIME Server.

This is an advanced topic and is covered in the KNIME Server Advanced Setup Guide.

Token-based authentication

KNIME Server also allows authentication by JWT (JSON Web Tokens) that have previously been issued by the server. The REST endpoint `/rest/auth/jwt` can be used to acquire such a JWT for the currently logged in user. Subsequent requests need to carry the token in the `Authorization` header as follows:

```
Authorization: Bearer xxx.yyy.zzz
```

where `xxx.yyy.zzz` is the JWT. Token-based authentication is enabled by default and cannot be disabled. However, you can restrict the maximum lifetime of JWTs issued by the server via the server configuration option `com.knime.server.login.jwt-lifetime`, see section KNIME Server configuration file options.

The OpenAPI documentation for the REST API which can be found at: `https://<hostname>/knime/rest/doc/index.html#/Session` should be considered the definitive documentation for this feature.

Large number of users in a group

Since the JWT includes the group membership for the user, this can get very large in some cases. JWTs with more than 30 groups and that are larger than 2kB are now compressed. If

they are still larger than 7kB a warning is logged with hints how to resolve potential problems.

One solution is to increase the maximum HTTP header size in Tomcat by adding the attribute maxHttpHeaderSize="32768" to all defined Connectors in the server.xml (the default is 8kB). In case Tomcat is running behind a proxy, the limit may need to be increased there, too. In case of Apache it's the global setting LimitRequestFieldSize 32768.

Database-based authentication

Database-based authentication is recommended to be used by small workgroups who do not have access to an LDAP system, or larger organisations in the process of trialing KNIME Server. If using the previously described H2 database it is possible to use the AdminPortal to manage users and groups. It is possible to use other SQL databases e.g. PostgreSQL to store user/group information, although in this case it is not possible to use the AdminPortal to manage users/groups, management must be done in the database directly.

For default installations this authentication method is enabled within the `server.xml` file. No configuration changes are required. In order to add/remove users, or create/remove groups the administration pages of the WebPortal can be used. The administration pages can be located by logging into the WebPortal as the admin user, see section Administration pages for more details.

Batch insert/update of usernames and roles is possible using the admin functionality of the KNIME Server REST API. This is described in more detail in the section RESTful webservice interface. A KNIME Workflow is available in the distributed KNIME Server installation package that can perform this functionality.

File-based authentication

For KNIME Server versions 4.3 or older the default configuration used a file-based authentication which we describe for legacy purposes. It is now recommended to use either database-based or LDAP authentication. The advantages of each are described in the corresponding sections above and below.

The XML file `<tomee-folder>/conf/tomcat-users.xml` contains examples on how to define users and groups (roles). Edit this file and follow the descriptions. By default this user configuration file contains the passwords in plain text. Encrypted storage of passwords is described in the Tomcat documentation.

Configuring a license server

Since version 4.3 KNIME Server can distribute licenses for extensions to the KNIME Analytics Platform (e.g. Personal Productivity, TeamSpace, or Big Data Connectors) to clients. In order to use the license server functionality, you require a master license. Every KNIME Server automatically comes with TeamSpace client licenses for the same number of users as the server itself. TeamSpace client licenses also cover all Personal Productivity features (such as Workflow Diff).

The master license file(s) should be copied into the `licenses` folder of the server repository (next to the server's license). The server will automatically pick up the license and offer them to clients. For configuring the client, see the section about "Retrieving client licenses" in the KNIME Explorer User Guide.

Client licenses distributed by the server are stored locally on the client and are tied to the user's operating system name (not the server login!) and its KNIME Analytics Platform installation and/or the computer. They are valid for five days by default which means that the respective extensions can be used for a limited time even if the user doesn't have access to the license server.

If the user limit for a license has been reached, no further licenses will be issued to clients until at least one of the issued licenses expires. The administrator will also get a notification email in this case (if their email notification is configured, see previous section Email notification).

License renewal

If the server is not behaving as expected due to license issues, please contact KNIME by sending an email to support@knime.com or to your dedicated KNIME support specialist.

If the license file is missing or is invalid a message is logged to the server's log file during server start up. KNIME clients are not able to connect to the server without a valid server license. Login fails with a message "No license for server found".

If the KNIME Server license has expired connecting clients fail with the message "License for enterprise server has expired on …". Please contact KNIME to renew your license.

If more users than are licensed attempt to login to the WebPortal, some users will see the message: "Maximum number of WebPortal users exceeded. The current server license allow at most <number of licensed users> WebPortal users.". In this case you will need to email KNIME at support@knime.com to discuss options to increase the number of licensed users.

After you receive a new license file, remove the old expired license from the `<server-repository>/licenses` folder. In case there are multiple license files in this folder, find the one containing a line with

```
"name" = "KNIME Server"
```

and the `"expiration date"` set to a date in the past. The license file is a plain text file and can be read in any text editor.

Store the new license file in the license folder with the same owner and the same permissions as the old file.

The new license is applied immediately; a server restart is not necessary.

Backup and recovery

The following files and/or directories need to be backed up:

- The full server repository folder, except the `temp` folder
- The full TomEE folder
- In case you installed your own molecule sketcher for the KNIME WebPortal (see above), also backup this folder.

A backup can be performed while the server is running but it's not guaranteed that a consistent state will be copied as jobs and the workflow repository may change while you are copying files.

In order to restore a backup copy the files and directories back to their original places and restart the server. You may also restore to different location but make sure to adjust the paths in the start script, the repository location in the context configuration file, and paths in the server configuration.

KNIME executor installation

Install the open-source KNIME Analytics Platform 4.1 on the server. Install all additional extensions users may need to run their workflows on the server. Make sure to include the "KNIME Report Designer" extension. Also install all extensions listed in the "KNIME Server Executor" category, either from the default online update site that or from the update site archive that you can get from the download area there. **Note that the versions of the KNIME Server Executor extensions must match the server's version (e.g. "4.10")!** Therefore, please check that you are installing from these extensions from correct update sites if you are not using the latest released versions of both the server and executor.

The easiest way to achieve this is to download the "KNIME + all free extensions" package from the public download page and extract it. It includes all extension required for running as an executor for a KNIME Server.

KNIME Analytics Platform must be executable for the server user (the user that runs the application server process, e.g. "tomcat"). If you use per-user KNIME executors, every user must be able to execute it on the server.
Make sure that users other than the installation owner either have no write permissions to the installation folder at all or that they have full write permission to at least the "configuration" folder. Otherwise you may run into strange startup issues. We strongly recommend revoking all write permissions from everybody but the installation owner.

If the server does not have internet access, you can download zipped update sites (from the commercial downloads page) which contain the extensions that you want to install. Go to the KNIME preferences at *File → Preferences → Install/Update → Available Software Sites* and add the zip files as "Archives". In addition you need to disable all online update sites on the same page, otherwise the installation will fail. Now you can install the required extensions via *File → Install KNIME Extensions…*.

Installing additional extensions

The easiest way to install additional extensions into the executor (e.g. Community Extensions or commercial 3rd party extensions) is to start the executor in GUI mode and install the extensions as usual. In case you don't have graphical access to the server you can also install additional extensions without a GUI. The standard `knime` executable can be started with a different application that allows changing the installation itself:

```
./knime -application org.eclipse.equinox.p2.director -nosplash
  -consolelog -r _<list-of-update-sites>_ -i _<list-of-features>_ -d _<knime-
installation-folder>_
```

Adjust the following parameters to your needs:

- `<list-of-update-sites>`: a comma-separated list of remote or local update sites to use. ZIP files require a special syntax (note the single quotes around the argument). Example:

  ```
  -r 'http://update.knime.org/analytics-
  platform/4.1,jar:file:/tmp/org.knime.update.analytics-platform_4.1.0.zip!/'
  ```

- `<list-of-features>`: a comma-separated list (spaces after commas are not supported) of features/extensions that should be installed. You can get the necessary identifiers by looking at *Help → About KNIME → Installation Details → Installed Software* in a KNIME instance that has the desired features installed. Take the identifiers from the "Id" column and make sure you don't omit the `.feature.group` at the end (see also screenshot on the next page). Example:

  ```
  -i org.knime.product.desktop,org.knime.features.r.feature.group
  ```

  You can get a list of all installed features with:

  ```
  ./knime -application org.eclipse.equinox.p2.director -nosplash \
    -consolelog -lir -d _<knime-installation-folder_
  ```

- `<knime-installation-folder>`: the folder into which KNIME Analytics Platform should be installed (or where it is already installed). Example:

  ```
  -d /opt/knime/knime_4.1
  ```

Updating the executor

Update of an existing installation can be performed by using the `update-rmi.sh` script in the root of the installation. You only have to provide a list of update sites that contain the new versions of the installed extensions and all installed extension will be updated (given that an update is available):

```
./update-rmi.sh http://update.knime.com/analytics-platform/4.1
```

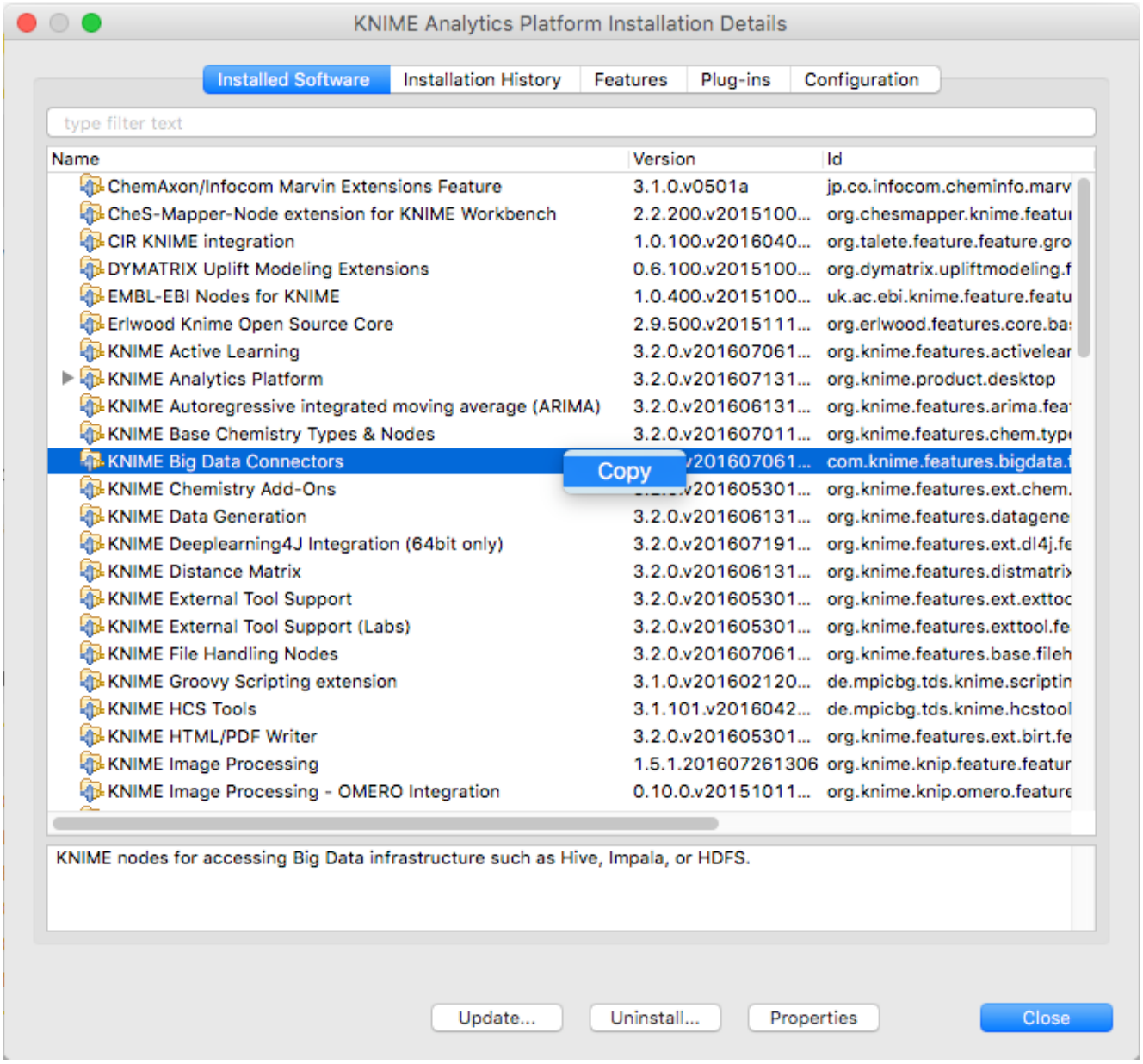
If you want to selectively update only certain extensions, you have to build the update command yourself. An update is performed by uninstalling (`-u`) and installing (`-i`) an

extension at the same time:

```
./knime -application org.eclipse.equinox.p2.director -nosplash -consolelog -r <list-of-update-sites> -i <list-of-features> -u <list-of-features> -d <knime-installation-folder>
```

To update the Big Data Extensions, for example, run the following command:

```
./knime -application org.eclipse.equinox.p2.director -nosplash \
  -consolelog -r http://update.knime.com/analytics-platform/4.1 -i
org.knime.features.bigdata.connectors.feature.group,org.knime.features.bigdata.spark.feature.group -u
org.knime.features.bigdata.feature.group,org.knime.features.bigdata.spark.feature.group
-d $PWD
```



Enabling workflow execution

Once you have installed the KNIME Executor with all necessary extensions, you have to tell the server where to find the executor. Set the value of `com.knime.server.executor.knime_exe` in the server configuration to the `knime` executable. The path can be absolute or relative to the server's configuration folder (`<server-repository>/config`). The path to the executor can be changed while the server is running it will be used when a new executor should be started (e.g. when the first workflow is being loaded).

For Windows users: For paths in the server configuration file either use forward slashes ("/") or double backslashes ("\\"). A single backslash is used to escape characters.

Sometimes workflow jobs running in the executor want to access files on the server, e.g. via workflow-relative URLs or by a URL using the server's mount point ID. Since the executor cannot authenticate itself to the server with the user's password (because it's generally not known by neither the server nor the executor) a token is generated by the server, when the workflow is started (or scheduled). This token represents the user including his group membership *at the time it is created*. If group membership changes while the workflow job is still running or there are further scheduled executions, these changes will not be reflected in the workflow execution. Also if access has been revoked from the user completely, existing (scheduled) jobs can still access the server repository.

If the executor is running on a different computer than the server, please pay attention to the following: The communication between server and executor is partially performed via the REST interface, e.g. when a workflow requests files from the server repository. Therefore the executor must know the server's address. The server tries to auto-detect its address and sends it to the executor. However, if the server is running behind a proxy (e.g. Apache) or has a different external IP address than internally, auto-detection will give a wrong address and the executor will not be able to reach the server. In this case you have to set the configuration option `com.knime.server.canonical-address` to the server's canonical address, e.g. `http://knime-server.behind.proxy/` (you do not need to provide the path to the server application). This address must be usable by the executor.

Per-user KNIME executors

If you install KNIME Server on a Linux or macOS operating system, you can configure KNIME Server to either use a single, global executor or multiple user specific executors to run your workflows. For Windows based installations, only the global execution mode is supported.

Running with multiple executors complicates the installation process and can be much less efficient depending on your degree of concurrent access to the server since each executor

has some computational overhead to maintain it. In order to mitigate this effect, you can limit the number of concurrent KNIME instances. Be aware that this will block users from executing workflows if the maximum is exceeded until a user frees his instance. Additionally, there is a preference in KNIME which can limit the number of threads used per executor.

The benefit of using multiple executors is that the workflows from different users run in relative isolation from one another; they don't share memory and should one executor become unresponsive, this should not directly affect the workflows running in the other executors. Additionally, in this case each executor is run by its user, which may be useful if your users will want to access resources on the system which are external to KNIME but are available to them at the user level. Finally, if you have KNIME Cluster Execution installed at the server and if you are using multiple executors, users will submit jobs to the cluster as themselves rather than a generic user.

By default the servers uses one KNIME Executor for all workflow jobs. On Linux systems KNIME Server offers the option to create a separate, KNIME executor instance for each user. To use multiple KNIME Executors, the following configuration is recommended:

1. Install the `sudo` package if necessary and add the following configuration by running `visudo` as root:

```
Cmnd_Alias KNIME_EXE = <path to KNIME executable>
knime-server ALL=(ALL) NOPASSWD: KNIME_EXE
#Defaults requiretty (needs to be commented out, i.e. disabled)
```

2. Set the following options in the server configuration:
    - `com.knime.server.sudo_cmd=<path to the sudo executable>` sets the path to the `sudo` executable (e.g. `/usr/bin/sudo`). If this option points to a non-existing file or is empty only one shared KNIME executor is used for all users.
    - `com.knime.server.rmi_port=<port>` sets the port the first KNIME executor instance listens to. The second instance listens to `<port>+1` and so on. Default value is `50100`.
    - `com.knime.server.rmi_max_instances=<n>` sets the maximum number of KNIME executor instances that are started. Please note that this parameter restricts the maximum number of users that can execute workflows on the server at the same time.

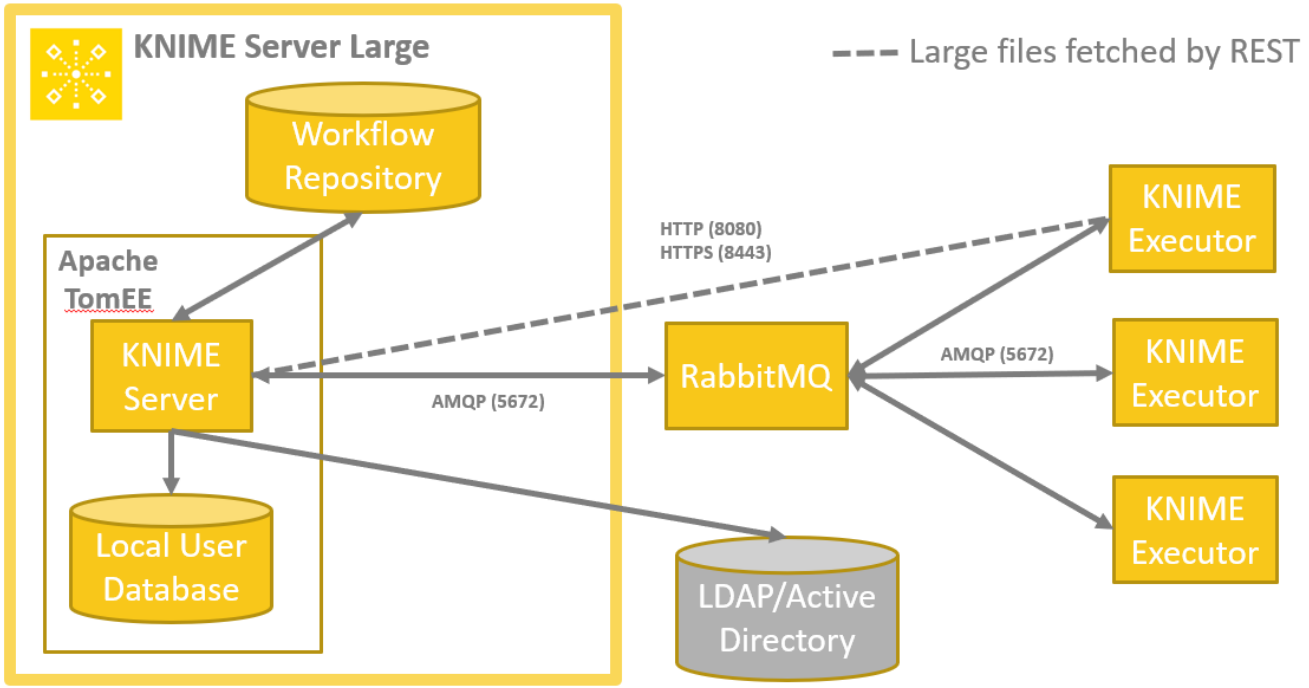
KNIME Server Distributed Executors

Distributed executors: Introduction

As part of a highly available architecture, KNIME Server 4.10 allows you to distribute execution of workflows over several executors that can sit on separate hardware resources. This allows KNIME Server to scale workflow execution with increasing load because it is no longer bound to a single computer. KNIME Server 4.10 implements the full functionality of the RMI executors.

If you're planning to use the distributed executors in production environments please get in touch with us directly, for more information.

Installation, configuration, and operation is very similar to the single executor setup. The server communicates with the executors via a message queueing system (and HTTP(S)). We use RabbitMQ for this purpose, and it's recommended, although not required, to install that on a separate machine as part of a highly available architecture.



Distributed executors: Installation instructions

Enabling distributed executors consists of the following steps:

- Install a new KNIME Server, following the KNIME Server Installation Guide.
- Shut down the server if it has been started by the installer.

- Install RabbitMQ following the instructions below.
- Adjust configuration files for the server and executor following the instructions below.
- Start the server and one or more executors.

Installing RabbitMQ

The server talks to the executors via a message queueing system called RabbitMQ. This is a standalone service that needs to be installed in addition to KNIME Server and the executors. You can install it on the same computer as KNIME Server or on any other computer directly reachable by both KNIME Server and the executors.

KNIME Server requires RabbitMQ 3.6+ which can be installed according to the Get Started documentation on their web page.

Make sure RabbitMQ is running, then perform the following steps:

- Enable the RabbitMQ management plug-in by following the online documentation
- Log into the RabbitMQ Management which is available at `http://localhost:15672/` (with user *guest* and password *guest* if this is a standard installation)
- Got to the *Admin* tab and add a new user, e.g. *knime*.
- Also in the *Admin* tab add a new virtual host (select the virtual hosts section on the right), e.g. using the hostname on which KNIME Server is running or simply *knime-server*.
- Click on the newly created virtual host, go to the *Permissions* section and set permission for the new *knime* user (all to ".*" which is the default).

Connecting Server and executor

KNIME Server and the executors now need to be configured to connect to the message queue.

For KNIME Server you must specify the address of RabbitMQ instead of the path to the local executor installation in the `knime-server.config`. I.e. comment out the `com.knime.server.executor.knime_exe` option (with a hash sign) and add the option `com.knime.enterprise.executor.msgq`. The latter takes a URL to the RabbitMQ virtual host: `amqp://<user>:<password>@<rabbit-mq-host>/<virtual host>`, e.g.

```
com.knime.enterprise.executor.msgq=amqp://<username>:<password>@rabbitmq-host/knime-server
```

Note that any special characters in the password must be URL encoded.

The same URL must also be provided to the executor as system property via the `knime.ini`:

```
-Dcom.knime.enterprise.executor.msgq=amqp://<username>:<password>@rabbitmq-host/knime
-server
```

Alternatively you can provide the message queue address as an environment variable:

```
KNIME_EXECUTOR_MSGQ=amqp://<username>:<password>@rabbitmq-host/knime-server
```

In case RabbitMQ High Available Queues are used, simply add additional `<rabbitmq-host>:<port>` separated by commas to the initial `amqp` address (this is supported with KNIME Server 4.10.5 and 4.11.3 onward):

```
-Dcom.knime.enterprise.executor.msgq=amqp://<username>:<password>@rabbitmq-host/knime
-server,amqp://<rabbitmq-host2>:<port2>,amqp://<rabbitmq-host3>:<port3>
```

In order to use RabbitMQ, you need to explicitly deactivate the embedded Qpid message broker by setting `com.knime.enterprise.executor.embedded-broker=false` in `knime-server.config`. Qpid does not support more than one KNIME Executor, and it doesn't support Executors running on separate hosts.

While commands between the server and KNIME Executors are exchanged via the message queue, actual data (e.g. workflows to be loaded) are exchanged via HTTP(S). Therefore, the KNIME Executors must know where to reach the server. The server tries to auto-detect its own address however in certain cases this address is not reachable by the executors or — in case of https connections — the hostname doesn't match the certificate's hostname. In such cases you have to specify the correct public address in the `knime-server.config` with the option `com.knime.server.canonical-address`, e.g.

```
com.knime.server.canonical-address=https://knime-server:8443/
```

You don't have to specify the context path as this is reliably auto-detected. Now you can start the server.

The executors must be started manually, the server does **not** start them. In order to start an executor (on any machine) launch the KNIME application (that has been created by the installer) with the following arguments:


```
./knime -nosplash -consolelog -application
com.knime.enterprise.slave.KNIME_REMOTE_APPLICATION
```

You can also add these arguments at the top of the `knime.ini` if the installation is only used as an executor. You can start as many executors as you like and they can run on different hosts. They will all connect to RabbitMQ (you can see them in the RabbitMQ Management in the *Connections* tab).

When you start the executor in a shell, a very simple command line interface is available to control the executor. Enter `help` at the `Executor>` prompt to get a list of available commands.

On Windows a separate window is opened for the executor process. In case there is a problem during startup (e.g. the executor cannot acquire core tokens from the server) then this window closes immediately. In this case you can add `-noexit` to the command above to keep it open and look at the log output or open at the log file which by default is `<user home>/knimeworkspace/.metadata/knime/knime.log` unless you provided a different workspace location with `-data`.

You may find it helpful for an executor to use customization profiles provided by the KNIME Server. In this case consult the documentation section for Customizations. For example editing the startup command for the executor will apply the `executor` profile.

```
./knime -nosplash -consolelog -profileLocation http://knime-
server:8080/knime/rest/v4/profiles/contents -profileList executor
com.knime.enterprise.slave.KNIME_REMOTE_APPLICATION
```

Running executors as services

It's also possible to run executors as services that are automatically started during system startup (and stopped during shut down). This is the recommended method to use when not running on a docker deployment.

Linux with systemd

Running executors as services is only supported on Linux distributions that use *systemd* (e.g. Ubuntu >= 16.04, RHEL 7.x and derivates). The following steps assume that you have a KNIME executor installed that contains the *KNIME Executor connector* extension as described in the section KNIME executor installation.

1. Copy the whole folder


```
<knime-installation>/systemd/
```

to the root of your file system. The folder includes the systemd service description for knime-executor and an override file that allows configuration of the service (such as file system location or the userid under which the executor should run).

2. Run

```
systemctl daemon-reload
```

3. Run

```
systemctl edit knime-executor.service
```

Adjust the settings in the editor that will open, and save the changes. Make sure that the `User` specified in this file exists on the system. Otherwise startup will fail unless your version of *systemd* supports `DynamicUser`. In this case a temporary user account will be created.

4. Enable the service with

```
systemctl enable knime-executor.service
```

Windows

On Windows executors can be run as Windows services by using *NSSM* (Non-Sucking Service Manager). The following steps assume that you have a KNIME Analytics Platform installation that contains the *KNIME Executor connector* extension as described in KNIME Server Installation Guide.

1. Edit

```
<knime-installation>/install-executor-as-service.bat
```

and adjust the variables at the top of the file to your needs.

2. Run this batch file **as administrator**. This will install the service.
3. Open the Windows *Services* application, look for the *KNIME Executor* service in the list and start it.

4. If you want to remove the executor service again, run the following **as administrator**:

```
<knime-installation>/remove-executor-as-service.bat
```

Note that if you move the KNIME Executor installation you first have to remove the service **before** moving the installation and then re-create it.

Load throttling

If too many jobs are sent to executors this may overload them and all jobs running on that executor will suffer and potentially even fail if there aren't sufficient resources available any more (most notably memory). Therefore an executor can reject new jobs based on its current load. By default an executor will not accept new jobs any more if its memory usage is above 90% (Java heap memory, averaged over 1-minute) or the average system load is above 90% (averaged over 1-minute). These settings can be changed by two system properties in the executor's `knime.ini` file:

Some options can be set as property in the `knime.ini` file as well as by defining an environment variable (Env). The environment variable changes will only take effect after a restart of the KNIME Executor. If the environment variable for an option is set, the property in the 'knime.ini' file will be ignored.

`-Dcom.knime.enterprise.executor.heapUsagePercentLimit=<value-in-percent e.g. 90>`

The average Heap space usage of the executor JVM over one minute. Default 90 percent

Env: `KNIME_EXECUTOR_HEAP_USAGE_PERCENT_LIMIT=<value-in-percent e.g. 90>`

`-Dcom.knime.enterprise.executor.cpuUsagePercentLimit=<value-in-percent e.g. 90>`

The average CPU usage of the executor JVM over one minute. Default 90 percent.

Env: `KNIME_EXECUTOR_CPU_USAGE_PERCENT_LIMIT=<value-in-percent e.g. 90>`

If only one distributed executor is available it will accept every job despite the defined Heap space and CPU limits. With KNIME Server 4.9.0 and later an option to change this behavior has been added. For more information see the Automated Scaling section.

Resource throttling

It is possible to restrict the number of cores/threads used by the Executor. In normal operation, you do not need to set this preference. Typically, the JVM will determine how many cores are available in the system (including identifying hyper-threaded cores as a 'core'), and the Executor will then set `knime.maxThreads=2*num_cores`.

In some cases, though, you may wish to restrict how many cores/threads the Executor can use. Examples of when this may be desired include when additional KNIME Executor cores on the machine must be reserved for another task, or in a local Docker setup where containers detect all cores available on a machine. Both of these configurations are typically not recommended, as it can be difficult to guarantee good resource sharing. Generally, it is better to run workloads on individual machines or in isolated pods using Kubernetes.

However, should you need to do so, you would use the following setting:

`/instance/org.knime.workbench.core/knime.maxThreads=<maximum number of threads to use>`

This setting controls the number of threads that the KNIME Executor will use to process workflows, and must be added to one of the preferences (`.epf`) files used by the Executor. (For more information on Executor preferences, see Executor Preferences.)

Automated Scaling

Currently we allow automated scaling by monitoring executor heap space and CPU usage, as well as the number of jobs running on an executor. It is also possible to blend these metrics using custom logic to invent custom scaling metrics. In some cases it may also be desirable to allow jobs to stack up on the queue and use the 'queue depth' as a fourth metric type. In order to do so, it is necessary to edit the knime.ini of the executors.

`-Dcom.knime.enterprise.executor.allowNoExecutors=<true|false>`

<Experimental Setting> Specifies whether the last executor accepting jobs is allowed to reject jobs. That will result in the behaviour that it is possible for jobs to pile-up on RabbitMQ. It may be necessary to increase the `com.knime.server.job.default_load_timeout` and the `com.knime.explorer.job.load_timeout` in the Analytics Platform to ensure sensible behaviour. The default is `false`, which emulates the behaviour before the setting was added.

When using an automatic scaling setup, jobs that are waiting for an executor to start, might run into timeouts. The default wait time for a job to be loaded by an executor can be increased by setting the `com.knime.server.job.default_load_timeout` option in the server configuration as described in section Server configuration files and options.

When starting jobs interactively using the Analytics Platform, the connection might also time out. The timeout can be increased by adding the following option to the knime.ini file of the KNIME Analytics Platform.

`-Dcom.knime.explorer.job.load_timeout=<duration with unit, e.g. 60m, 36h, or 2d>`

Specifies the timeout to wait for the job to be loaded. The default duration is `5m`.

Generally, the timeout in the Analytics Platform should be higher than the timeout set in the KNIME server. This prevents the interactive session from running into read timeouts.

Reconnecting to message queue

In case the connection to the message queue gets lost (e.g. by restarting RabbitMQ), starting with KNIME Server 4.11 the executor will try to reconnect to the message queue. The following option can be adjusted in the `knime.ini` file of the executor:

`-Dcom.knime.enterprise.executor.connection_retries=<number of retries>`

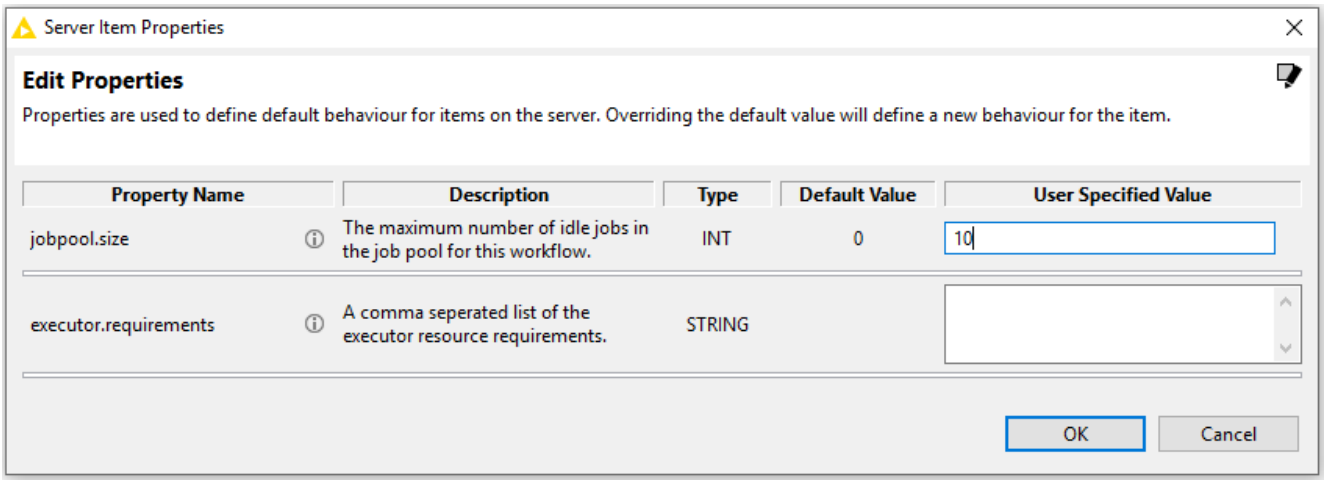
Specifies the number of retries that should be attempted to reconnect to the message queue. Between each attempt the executor waits 10 seconds. The default value is set to `9` i.e. the executor tries reconnecting for 90 seconds. Note that this option can be also set via the environment variable `KNIME_EXECUTOR_CONNECTION_RETRIES`, which takes precedence over the system property set in the `knime.ini` file.

Job Pools

For workflows that are frequently executed it's now possible (starting with KNIME Server 4.8.1) to keep a certain number of jobs from that workflow in memory. This eliminates the overhead of loading the workflow in an executor after the first use of that job. This should be particularly beneficial in cases where job loading time is large compared to job execution time.

Enabling job pools

In order to enable a job pool, a property has to be set on the workflow that should be pooled. Setting workflow properties can be done in the KNIME Explorer (starting with KNIME Server 4.9.0) by right-clicking on a workflow and selecting 'Properties…'. A dialog will open that lets the user view and edit the properties of the workflow.



Otherwise, workflow properties can also be set via a REST call, e.g. using `curl`:

```
curl -X PUT -u <user>:<password> http://<server-
address>/knime/rest/v4/repository/<workflow>:properties?com.knime.enterprise.server.jobp
ool.size=<pool size>
```

This will enable a pool with at most *pool-size* jobs for the workflow *workflow*.

It is only possible for single-call executions that do loading, execution, and discard in one call (i.e. the current `:execution` resource). Jobs that clients execute with multiple REST calls (load, execute, re-execute, discard) cannot be pooled.

Disabling job pools

Job pools can be disabled by setting the job pool size to 0, either in the KNIME Explorer or via a REST call:

```
curl -X PUT -u <user>:<password> http://<server-
address>/knime/rest/v4/repository/<workflow>:properties?com.knime.enterprise.server.jobp
ool.size=0
```

Using job pools

In order to make use of the pooled jobs, a special REST resource has to be called for executing a job. Instead of calling out to `:execution` you have to call to `:job-pool`. Apart from that both calls are identical concerning semantics and allowed parameters.

Executing a pooled job might look as follows:

```
curl -u <user>:<password> http://<server-
address>/knime/rest/v4/repository/<workflow>:job-pool?p1=v1&p2=v2
```

This will call *workflow* passing *v1* for input parameter *p1* and *v2* for input parameter *p2*. Calls using `POST` will work in a similar way using the `:job-pool` resource.

Behaviour of job pools

Job pools exhibit a certain behaviour which is slightly different from executing a non-pooled job. Clients should be aware of those differences.

- If the pool is empty (either intially or if all pooled jobs are currently in use) the job will be loaded from the workflow and thus the call will take longer.
- A used job will be put back into the pool right after the result has been returned if the pool isn't already full. Otherwise the job will be discarded.
- Pooled jobs are tied to the user that triggered initial loading of the job. A pooled job will never be shared among different users.
- If there is no job in the pool for the current user, the oldest job in the pool from a different user will be removed. This can lead to contention if there are more distinct users calling out to the pool than the pool size.
- Pooled jobs will be removed if they are unused for more than the configured job swap timeout (see the server configuration options).

- A pooled job **without** any input nodes will be reset before every invocation, even the first one! This is different from executing a non-pooled job but is required for consistent behaviour across multiple invocations. Otherwise the first and subsequent operations may behave differently if the workflow is saved with some executed nodes.
- In a pooled job **with** input nodes all of them will receive input values before execution: either the value that has been passed in the call, or if no explicit value has been provided its default value. This means that **all** input nodes will be reset prior to execution and not just the nodes explicitly set in the call. Again, this is different from executing a non-pooled job where only input nodes with explicitly provided values will be reset but required for consistency. Otherwise the results of a call may depend on the parameters passed in the previous call.

Workflow Pinning

Workflow pinning can be used to let workflows only be executed by a specified subset of the available executors when KNIME Server Distributed Executors are enabled.

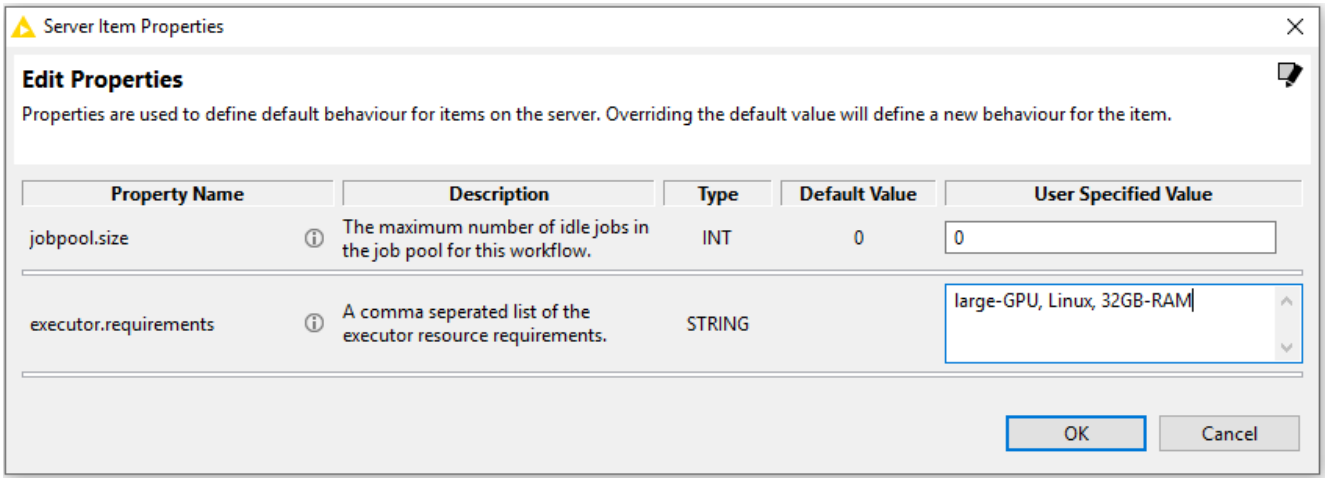
For workflows that need certain system requirements (e.g. specific hardware, like GPUs, or system environments, like Linux) it's now possible (starting with KNIME Server 4.9.0) to define such executor requirements per workflow. Only executors that fulfill the executor requirements will accept and execute the workflow job. To achieve this behaviour, a property has to be set for the workflows. Additionally, the system admin of the executor's has to specify a property for each executor seperately. The properties consist of values that define the executor requirements, set for a workflow, and executor resources, set for an executor, respectively.

Prerequisites for workflow pinning

In order to use workflow pinning, the KNIME Server Distributed Executors must be enabled and RabbitMQ must be installed. Otherwise, the set executor requirements are ignored.

Setting executor.requirements property for a workflow

Executor requirements for a workflow can be defined by setting a property on the workflow. The executor requirements are a simple comma-seperated list of user-defined values. Setting workflow properties can be done in the KNIME Explorer by right-clicking on a workflow and selecting 'Properties...'. A dialog will open that lets the user view and edit the properties of a workflow.



Alternatively, workflow properties can also be set via a REST call, e.g. using `curl`:

```
curl -X PUT -u <user>:<password> http://<server-
address>/knime/rest/v4/repository/<workflow>:properties?com.knime.enterprise.server.exec
utor.requirements=<executor requirements>
```

This will set the executor requirements *executor-requirements* for the workflow *workflow*.

Setting executor.resources property for an executor

To define which resources an executor can provide, a property has to be set for the executors. This can be done in two ways:

1. Setting an environment variable on the system of an executor. The name of the variable has to be 'KNIME_EXECUTOR_RESOURCES' and the value must be a comma-seperated list of user-defined values.

   ```
   KNIME_EXECUTOR_RESOURCES=value1, value2, value3
   ```

2. Setting a system property in the *knime.ini* file, which is located in the installation folder of the executor. The file contains the configuration settings of the executor, i.e. options used by the Java Virtual Machine. The name of the property has to be 'com.knime.enterprise.executor.resources' and the value must be a comma-seperated list of user-defined values.

   ```
   -com.knime.enterprise.executor.resources=value1, value2, value3
   ```

The environment variable has priority over the system property if both are specified.

Removing executor.requirements property for a workflow

Executor requirements can be removed by setting the property to an empty field. This can be done either in the KNIME Explorer or via a REST call:

```
curl -X PUT -u <user>:<password> http://<server-
address>/knime/rest/v4/repository/<workflow>:properties?com.knime.enterprise.server.exec
utor.requirements=
```

Removing executor.resources property for an executor

The property can be removed either by completely removing the environment variable or by completely removing the property in the *knime.ini* file depending on the way the property was set. Alternatively, the property can also be removed by leaving the value of the environment variable or the value of the property in the *knime.ini* file empty.

A restart of the executor is required to apply the changes.

Behaviour of executor requirements

An executor only accepts a job if it can fulfill all the executor requirements that were defined for the workflow. Otherwise, it will just ignore the job.

- Jobs with no executor requirements will be accepted by all available executors.
- The executor.requirements property values only need to be a subset of the executor's defined executor.resources property values in order for the workflow to be accepted by the executor for execution.
- If no executor can fulfill the executor requirements, the queued job will be discarded.
- If the appropriate executors cannot accept new jobs because their load is too high, the new queued job will run in a timeout (normally after 60 seconds) and discard itself, see Load throttling.

```
Example:
Workflow1 executor.requirements: medium_RAM, two GPU, Linux
Workflow2 executor.requirements: small-RAM, Linux
Workflow3 executor.requirements:
Executor1 executor.resources: small-RAM, Linux, two GPU
Executor2 executor.resources: medium_RAM, Windows, two GPU
Workflow1 will be ignored by both executors and will be discarded.
Workflow2 will be ignored by Executor2 and accepted by Executor1.
Workflow3 will be accepted by any of the available executors.
```

Execution lifecycle

During the course of executing (or running) a workflow, there are several things that happen. Most of the time you don't need to know about this, but sometimes in more complex deployments, or for detailed debugging it may be helpful to understand the lifecycle of a workflow that is executed.

Workflows, Jobs and Job states

Workflows

The workflow is the collection of nodes, setup to perform your data analysis task. A workflow will contain all of the relevant (default) settings to perform the analysis. In addition to the settings a workflow may contain some data, e.g. if the workflow has been partially executed locally and then uploaded to the KNIME Server. A more full description of a workflow, and how to create one is available create-your-first-workflow

Jobs

On the KNIME Server, a Job is created whenever a workflow is executed. A full copy of the workflow is made into a location where other workflow executions can't interfere with it. For full details see executing-a-workflow-on-the-server

Job states

Jobs exist in a variety of different states, which are displayed in either the Explorer view of the KNIME Analytics Platform, or the Jobs tab on the AdminPortal. The job states are:

- **UNDEFINED** - This is the first state of a job, and may be seen in the case where an executor cannot communicate with the server due to network issues, or the executor not having enough free CPU/RAM resources.
- **CONFIGURED** - The Job has executed to a certain point, and is waiting for user input e.g. waiting for WebPortal page input by the user clicking Next.
- **IDLE** - With the current configuration of the nodes, no further nodes can be executed. This is either because a scheduled workflow failed, or if the workflow is executed via the Webportal or via REST it might wait for input.
- **EXECUTING** - Job is currently executing.

- **EXECUTED** - Job has been executed (may still be in memory, see notes below)
- **DISCARDED** - Job has been executed and discarded (meaning executor resources, and server disk space are freed up.)

Note that in addition to the job states there is the `In Memory` flag. The flag tells us whether the workflow is residing in the executor memory, or has been swapped back to disk in the KNIME Server Repository. The setting `com.knime.server.job.max_time_in_memory` documented in KNIME Server configuration file options defines how long a job will remain in memory before being swapped. Additionally, when an executor is gracefully shutdown then all jobs currently in memory are swapped back to disk. Additionally it's possible to manually force a job to swap to disk by issuing a REST call via SwaggerUI for Workflows using the job UUID.

Remote Workflow Editor

Introduction

The KNIME Remote Workflow Editor enables users to investigate the status of jobs on the server. Whenever a workflow is executed on the KNIME Server, it is represented as a job on the server. This instance of your workflow will be executed on the KNIME Server, which can be helpful in cases where the server hardware is more powerful than your local hardware, the network connection to external resources such as databases is faster, and does not require traversing firewalls/proxies.

What is the Remote Workflow Editor

The Remote Workflow Editor looks just like your local workflow editor, apart from the fact that it is labelled and the canvas has a watermark to help identify that the workflow is running on the KNIME Server.

Most of the edit functionality that you would expect from editing a workflow locally on your machine is possible. Notable cases where it's not yet supported are: copying nodes from a local workflow to a remote workflow (and vice-versa), browse dialog for file reader/writer nodes browses the local filesystem rather than the remote filesystem.

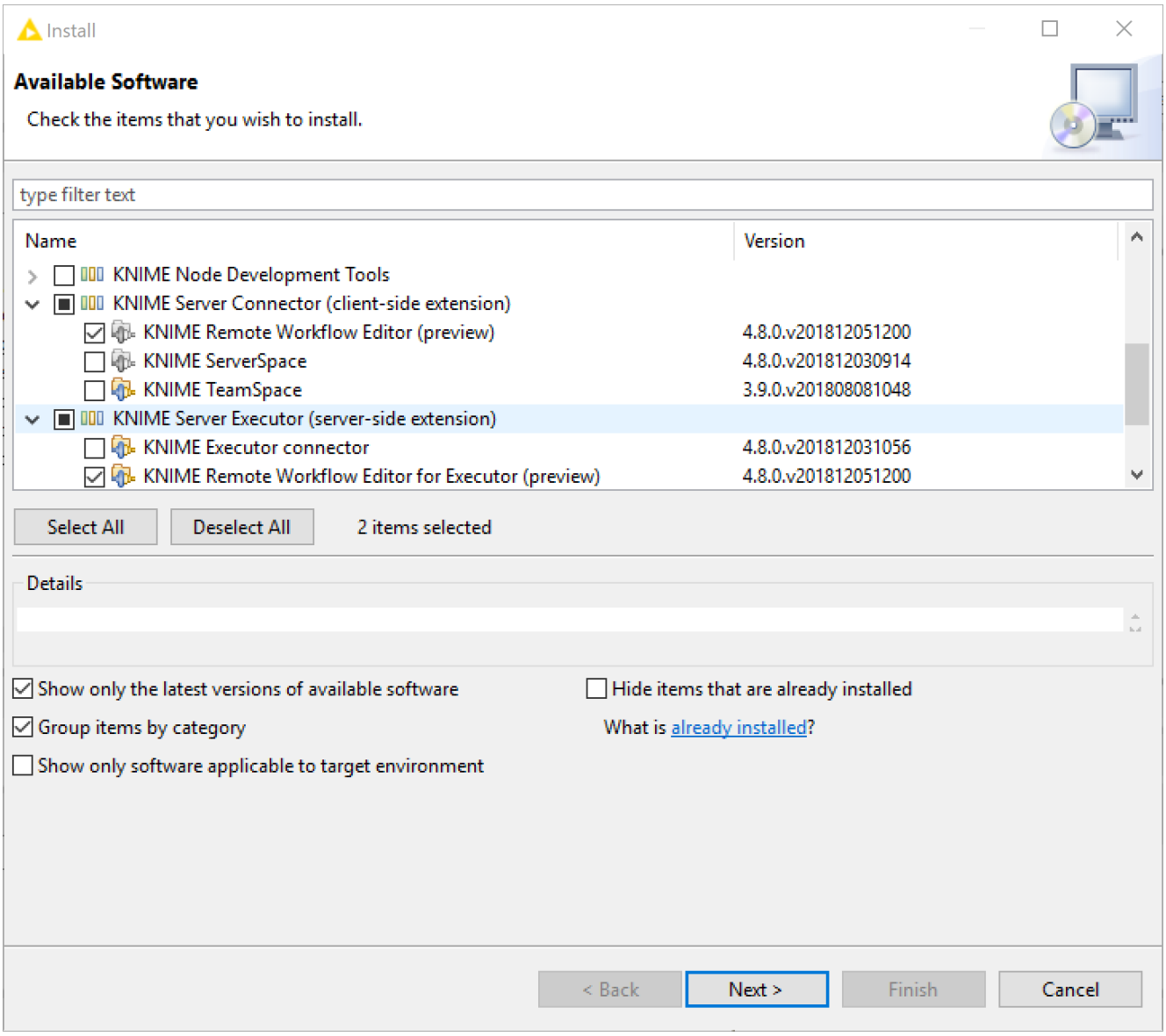
Installation

The Remote Workflow Editor is installed on the KNIME Analytics Platform as part of the KNIME Server Connector extension, and on the KNIME Server it must be installed into each executor. Detailed instructions are found below.

Server setup

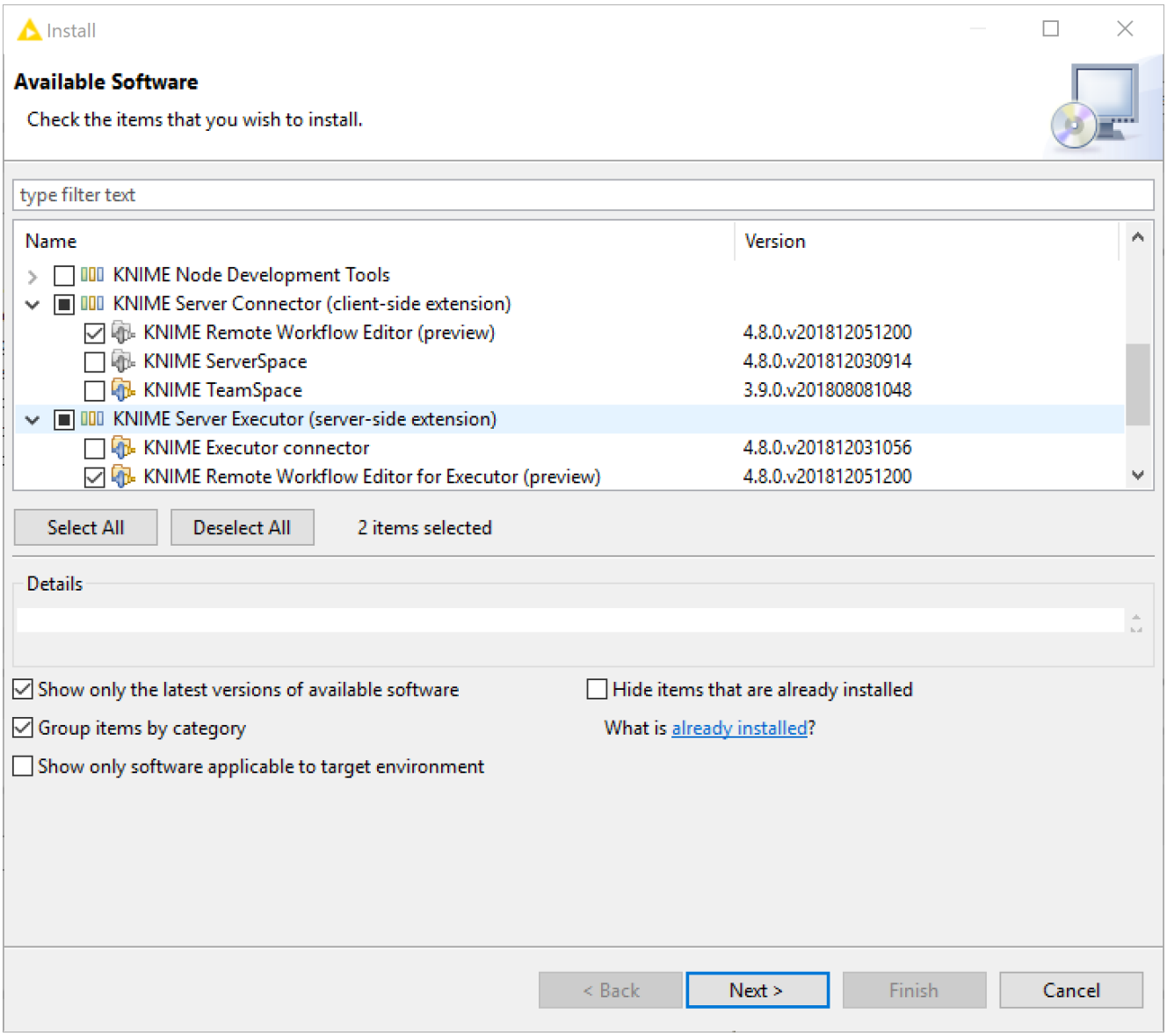
If KNIME Server is installed on Windows Server, then you may use the GUI to install the "KNIME Executor connector" from the "KNIME Server Executor (server-side extension)" feature. For Linux servers it is normally easier to use the command line to install the feature. Change to the KNIME Executor installation directory, and use the command:

```
./knime -application org.eclipse.equinox.p2.director -nosplash \
  -consolelog -r +https://update.knime.com/analytics-platform/{version_exe}+ -i \
  com.knime.features.gateway.remote.feature.group -d $PWD
```



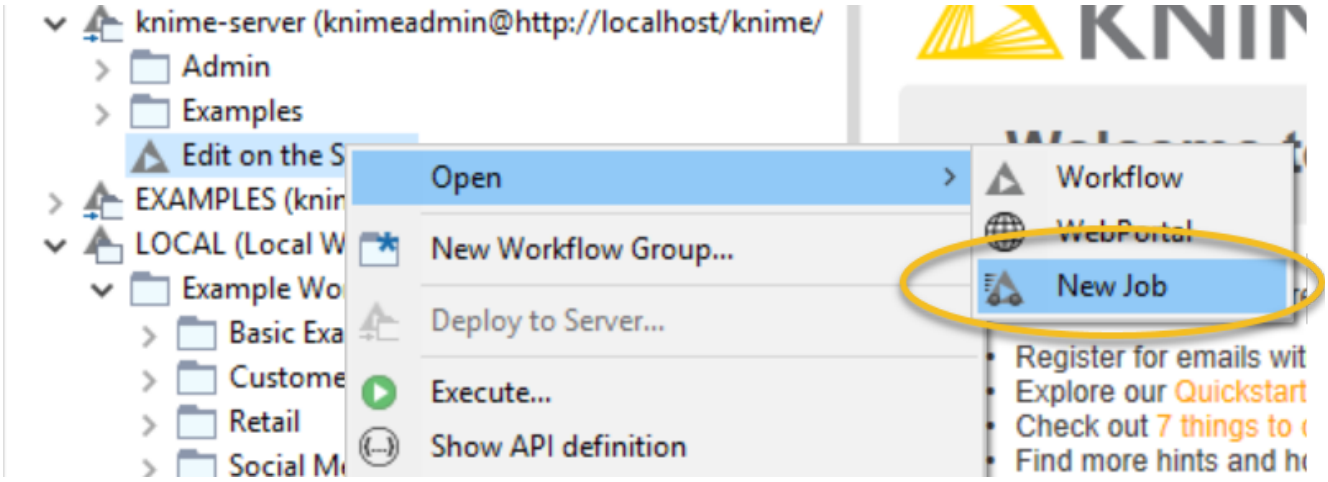
Analytics Platform setup

The Remote Workflow Editor feature needs to be installed in the KNIME Analytics Platform. Choose File > Install KNIME Extensions, and then select "KNIME Remote Workflow Editor" from the "KNIME Server Connector (client-side extension)" category.

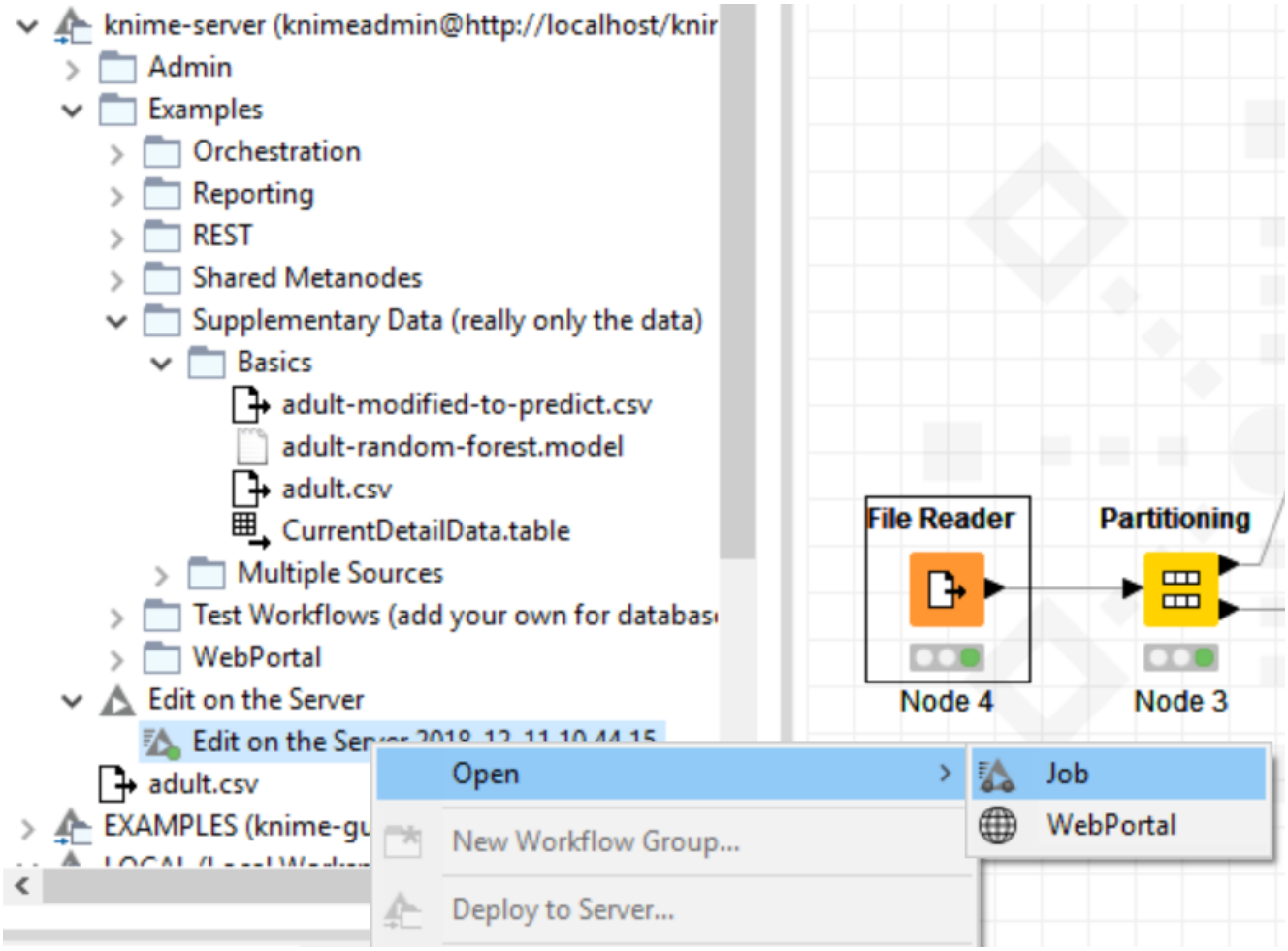


Usage

It’s possible to create and open a new job from a workflow residing on the KNIME Server, using the KNIME Analytics Platform, by using the 'Open Job' context menu. See:



Jobs that are already created, e.g. by using the Execute context menu, a scheduled job, or a job started in the WebPortal, can be visualized by selecting the job and using the 'Open Job' context menu. See:

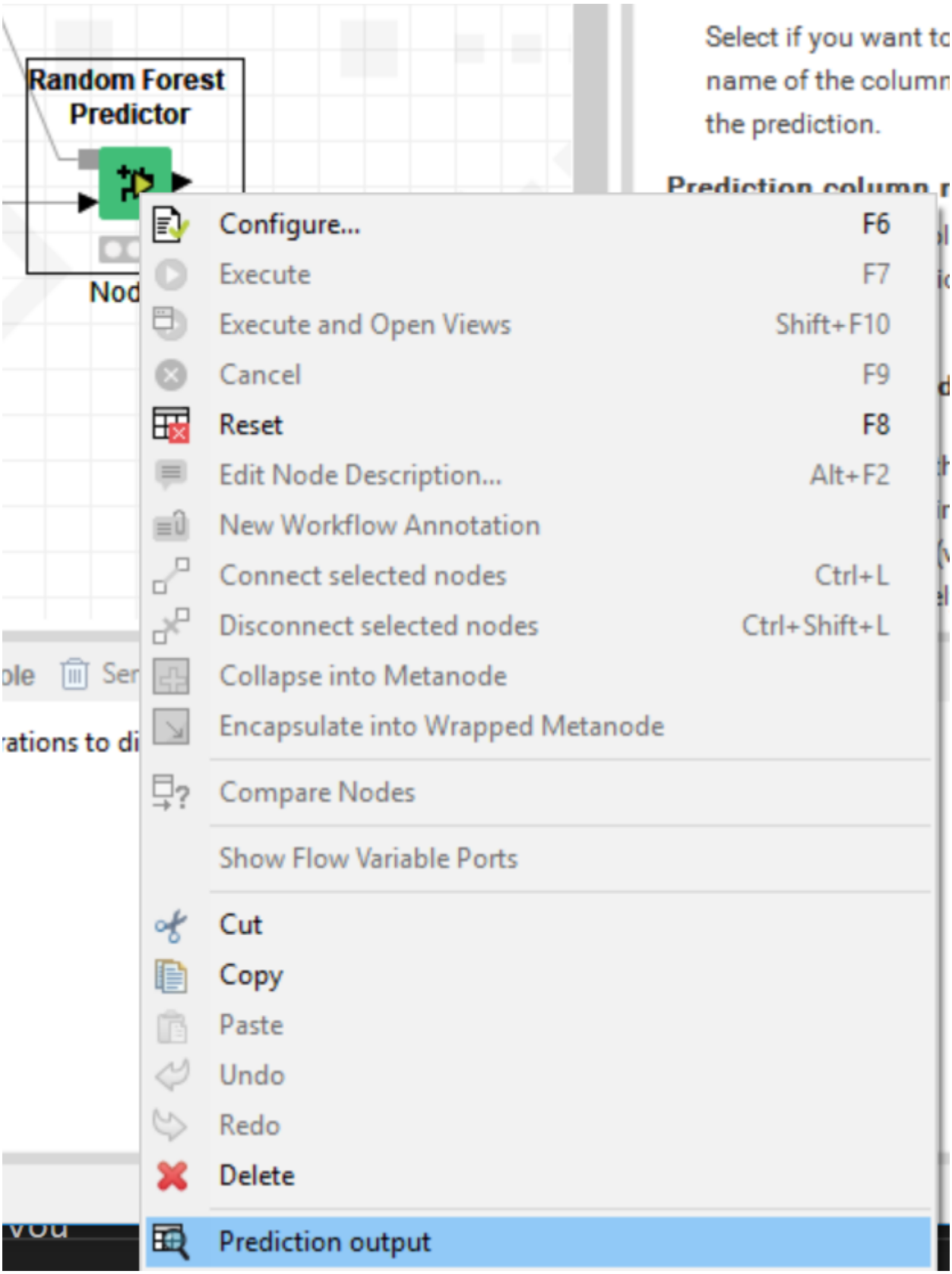


All jobs (executed from Analytics Platform, or via WebPortal) can be viewed, meaning that it's possible to see node execution progress, number of rows/columns generated, and any warning/error messages.

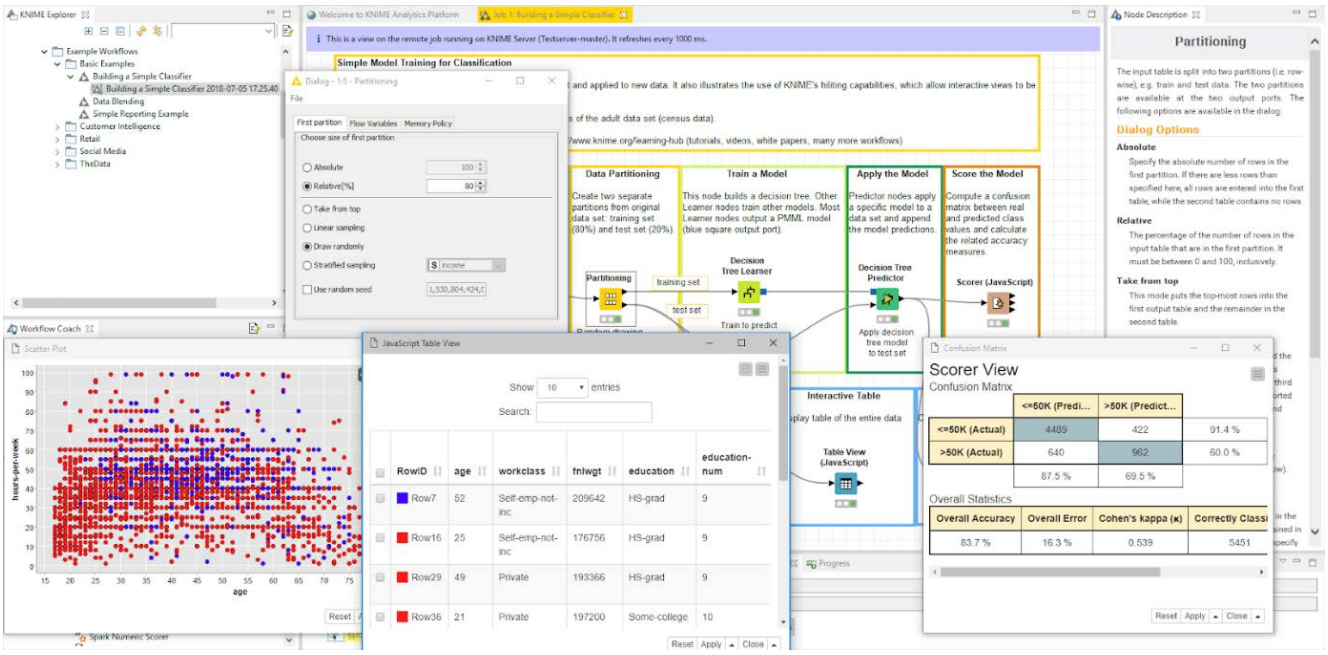
It's also possible to view and edit configuration settings, of most nodes as you would if the

workflow was on your local KNIME Analytics Platform. Currently it’s not supported to configure file paths in some file reader nodes.

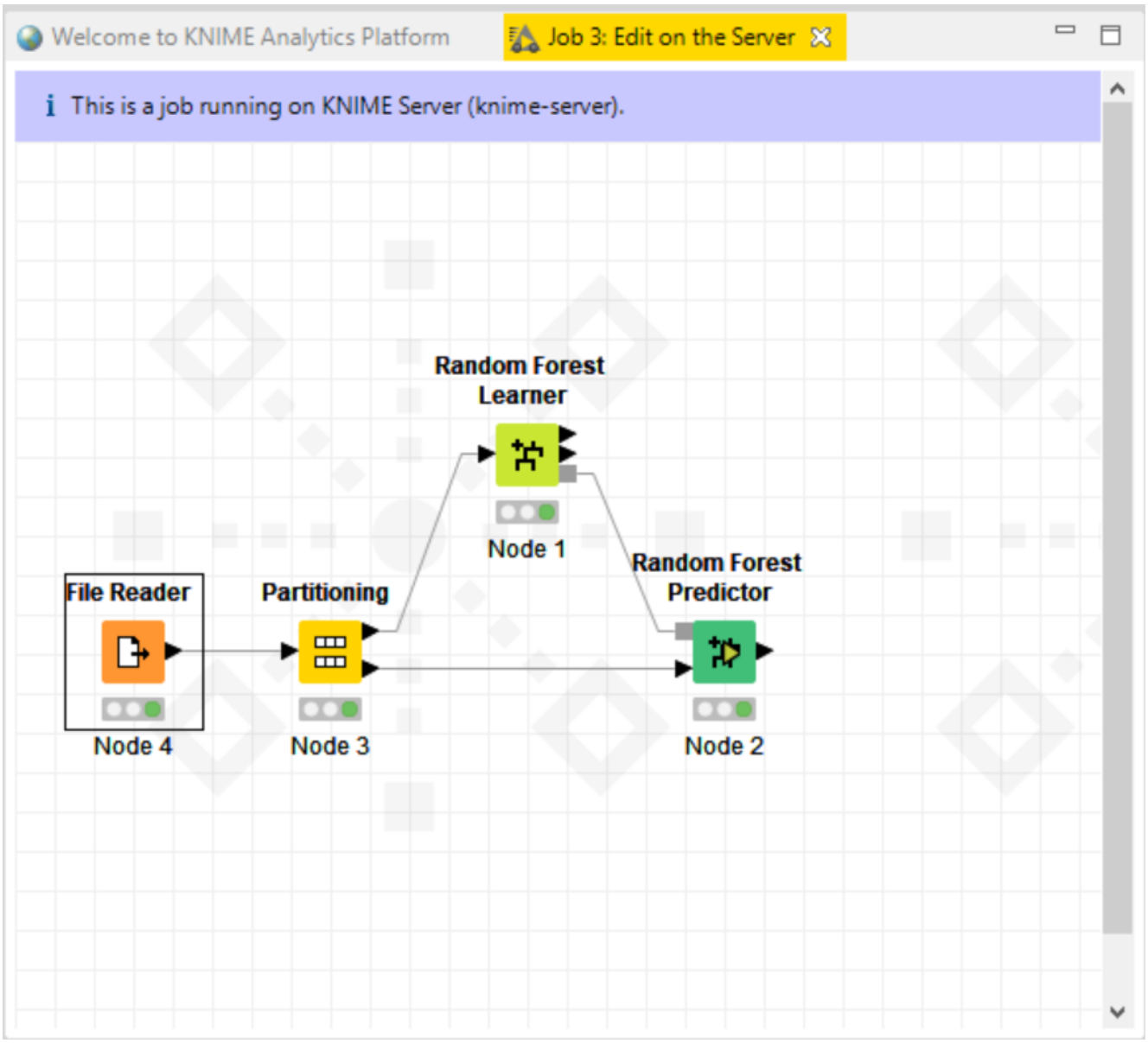
It’s also possible to move, add and remove nodes from the workbench, as you would for a local workflow.



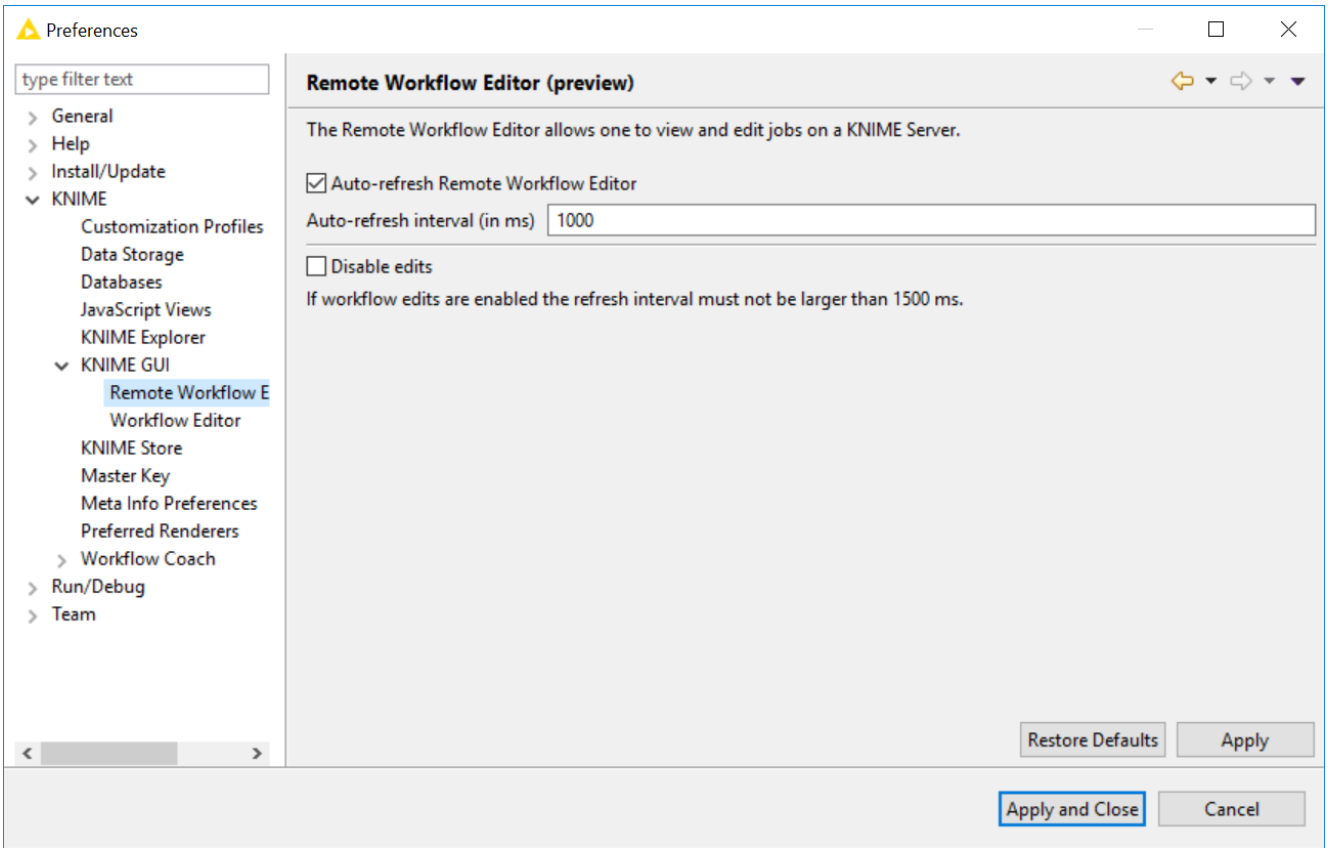
With the KNIME Server 4.8 release it's possible to view data via the normal data view.



It's possible to view data and views by using the JavaScript Views.



The KNIME Remote Workflow Editor enables you to view and edit workflow jobs on the KNIME Server.



Remote Workflow Editor preferences allow to change the auto-refresh interval, and also optionally to uncheck the 'Enable job edits' to enforce view only mode for all workflows.

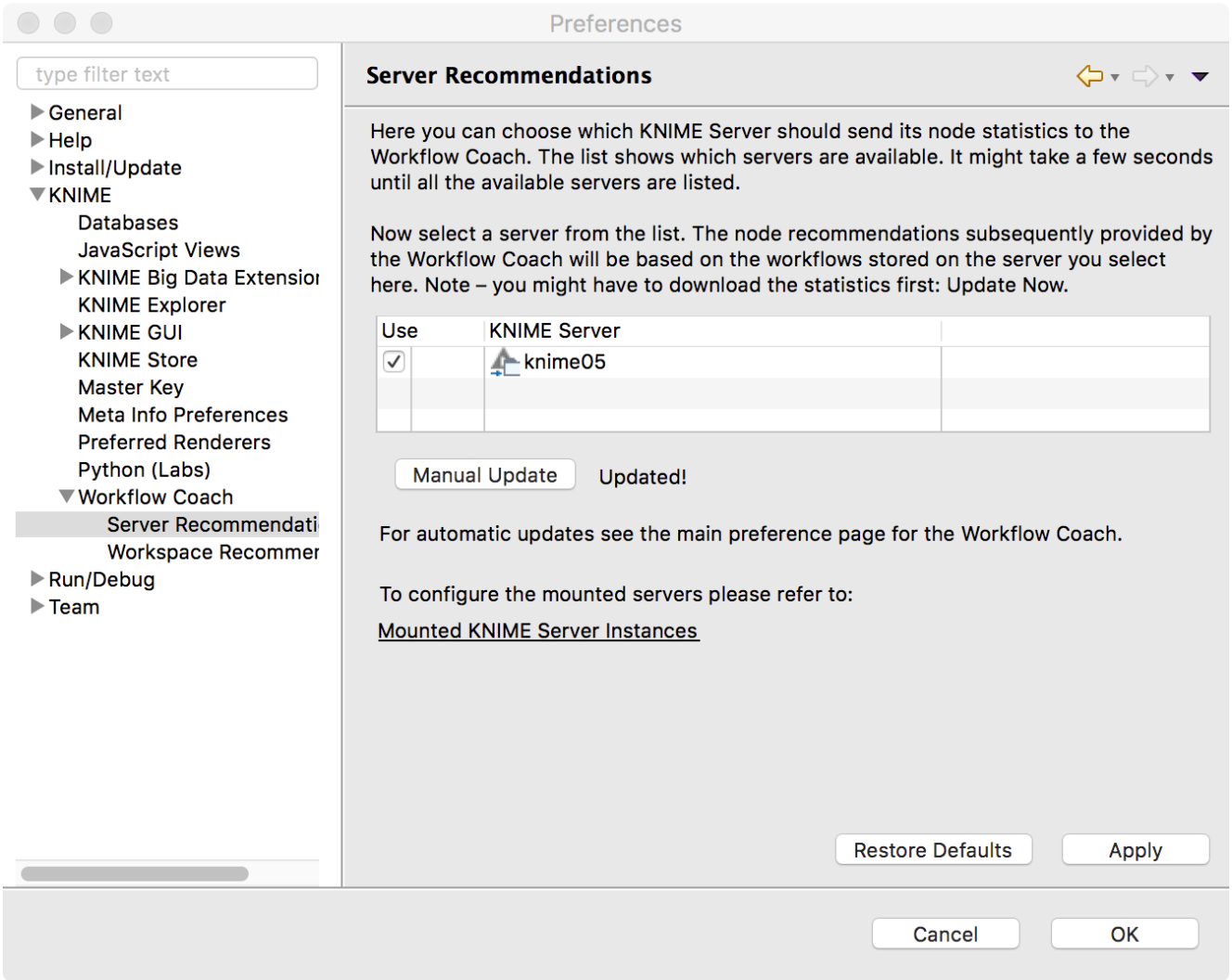
You will be able to see which nodes are currently executing, which are already executed, and which are queued to be the next in execution. You can see errors and warning in the workflow by mouseover on the respective sign.

Inside a failing job you can see the error and warning messages by mouse-over the respective sign

Custom Workflow Coach recommendations

KNIME Server is able to serve custom node recommendations to the workflow coach. In order to enable this functionality `com.knime.server.repository.update_recommendations_at=` must be set as described in the `knime-server.config` settings table.

The KNIME Analytics Platform preferences must be updated to enable the additional workflow coach recommendations:



Management Services for KNIME Analytics Platform: Customizations

Customizations allows to define centrally managed:

- Update sites
- Preference profiles (Database drivers, proxy settings, Python/R settings, etc.)

KNIME Server allows you to distribute customization profiles to connected KNIME Analytics Platform clients. A profile consists of a set of files that are fetched by the client during startup. The files are copied into the user's workspace. Files ending with `.epf` are treated as Eclipse preferences and can be used to override the *default* preferences which are usually defined by the extension vendors. Settings that an Analytics Platform user has already changed (i.e. which don't have the default value any more) are not affected. However, the user can choose to *"Restore ALL preferences to defaults"* via the preference page in the KNIME Analytics Platform. In this case the user is first prompted, then a backup of the preferences file is stored in the `<knime-workspace>/.metadata/knime/preferences-backup.epf`, finally, the server-managed settings will replace any preferences with the configured default values. The feature is available to all KNIME Server named-users and additionally to all registered consumers.

Analytics Platform Customization

The server installer will create a customization template profile in `config/client-profiles.template/customizations`. It consists of a preference file that contains all available configuration settings (including detailed descriptions) as well as some additional files that may be referenced in the preference file. Please see `customizations.epf` for details.

Server-side setup

In order to enable server-managed customization on the server side you have to create one or more subdirectories inside `<server-repository>/config/client-profiles`. New server installations already come with an example profile that you can use as a starting point. You can have an arbitrary number of profiles. Which profiles are fetched by the client and in which order is defined by settings in the client (see below). If more than one profile defines a preference value, the *last* profile in the list requested by the client will determine the actual default value. Let's have a look at an example.

Suppose the `config/client-profiles` folder on the server has the following contents:

```
.../config/client-profiles/base/base.epf
        org.knime.workbench.explorer.view/knime.explorer.mountpoint=...
          org.knime.workbench.ui/knime.maxThreads=4
.../config/client-profiles/base/my-db-driver.jar
.../config/client-profiles/linux/linux.epf
        org.knime.workbench.ui/knime.maxThreads=8
        org.knime.python2/python2Path=/usr/bin/python2
        org.knime.python2/python3Path=/opt/anaconda3/bin/knime-python
.../config/client-profiles/windows/windows.epf
        org.knime.python2/python3Path=C:/Anaconda3/bin/knime-python.bat
.../config/client-profiles/windows/my-lib.dll
.../config/client-profiles/windows/my-db-driver.jar
```

If the client request the profiles "base,linux" (in this order), the default number of threads used by KNIME nodes will be 8. The python paths are set to the correct Linux paths. If another client requests "base,windows" the default number of threads will be 4 (from the base profile) and the Python 3 path will be set to a folder on the `C:\` drive. The pre-defined KNIME Explorer mount points will be identical for both clients because the value is only defined in the base profile.

A profile may contain several preferences files. They are all virtually combined into a single preference file for this profile in alphabetical order.

A profile may contain additional resources, for example JDBC driver files. The entire contents of the client-profiles folder including hidden files is sent to the client as a zip file and unpacked into a location in the client workspace. There is no conflict handling for any other files in the requested profiles (e.g. `my-db-driver.jar`) because they will end up in separate subdirectories on the client and not be processed further.

For further details and an example on how to distribute JDBC driver files go to the Server-managed Customization Profiles section of the KNIME Database Extension Guide.

If KNIME Server is running on Linux or macOS then the permissions of files inside profiles are transferred to the clients. This is useful for executable files on Linux or macOS clients, such as shell scripts. If you have such files in your profiles make sure to set the permissions accordingly on the server. KNIME Server's running on Windows don't support this feature because Windows file systems don't have the concept of executable files.

Note that the profiles on the server are accessible without user authentication therefore they shouldn't contain any confidential data such as passwords.

In order to create preference files for clients, start a KNIME Analytics Platform with a *fresh workspace* on the desired environments (e.g. Linux, Windows). This ensures that all preferences are set to their vendor defaults. Then change the preferences to your needs and export them via *File → Export → KNIME Preferences*. Then copy the resulting `epf` file to the

profile folder on the server.

Variable replacement

It is possible to use variables inside the preferences files (only those files ending in `.epf`) which are replaced on the client right before they are applied. This makes the server-managed customizations even more powerful. These variables have the following format: `${prefix:variable-name}`. The following prefixes are available:

- **`env`**: the variable is replaced with the value of an environment value. For example, `${env:TEMP}` will be replaced with `/tmp` under most Linux systems.
- **`sysprop`**: the variable is replaced with a Java system property.
  For example, `${sysprop:user.name}` will be replaced with the current user's name. For a list of standard Java system properties see the JavaDoc. Additional system properties can be defined via `-vmargs` in the `knime.ini`.
- **`profile`**: the variable will be replaced with a property of the profile in which the current preference file is contained in. Currently "location" and "name" are supported as variable names. For example, `${profile:location}` will be replaced by the file system location of the profile on the client. This can be used to reference other files that are part of the profile, such as database drivers:
  `org.knime.workbench.core/database_drivers=${profile:location}/db-driver.jar`
- **`origin`**: the variable will be replaced with a HTTP response header sent by the server with the downloaded profiles. In addition to standard HTTP headers (which are probably not very useful), the following KNIME-specific origin variables are available:
  - `${origin:KNIME-Default-Mountpoint-ID}` — the server's configured default mount ID
  - `${origin:KNIME-EJB-Address}` — the address used by the KNIME Explorer; see the client profile templates in the repository created by the installer for an example
  - `${origin:KNIME-REST-Address}` — base address of the server's REST interface
  - `${origin:KNIME-WebPortal-Address}` — address of the server's WebPortal
  - `${origin:KNIME-Context-Root}` — base path on the server where all KNIME resources are available, usually `/knime`.
- **`custom`**: the variable will be replaced by the custom profile provider implementation that is also used to provide the profile location and list.

In case you want to have a literal in a preference value that looks like a variable, you have to use two dollar signs to prevent replacement. For example `$${env:HOME}` will be replaced with

the plain text `${env:HOME}`. If you want to have two dollars in plain text, you have to write three dollars (`$$${env:HOME}`) in the preference file.

Note that once you use variables in your preference files they are not standard Eclipse preference files anymore and cannot be imported as they are.

Client-side setup

The client has three possibilities to request profiles from a KNIME Server.

1. Two command line arguments which define the address and the (ordered) list of requested profiles (note that the command line argument and the variable must be separated onto two lines — as seen below):

   ```
   -profileLocation
   http://knime-server:8080/knime/rest/v4/profiles/contents
   -profileList
   base,linux
   ```

   Both arguments must be supplied either directly on the command line or in the `knime.ini` **before** the `-vmargs`.

2. Two preference settings in the "KNIME/Customization profiles" preference page. There the user can select a server and then define the ordered list of profiles that he/she wants to apply. Note that this setting cannot be controlled by the server-managed customization profiles. Changes will take effect after the next start.
3. A custom profile provider defined in a custom Eclipse plug-in. Since this involves writing Java code and is likely only of interest for large-scale installations we cover this approach in the KNIME Server Advanced Setup Guide.

The three possibilities are tried in exactly this order, i.e. if one of them provides a server address and a non-empty list of profiles it will be used and all following providers will be skipped.

It's also possible to provide a local file system folder as the `profileLocation` on the command line (or in your custom profile provider). The layout of this local folder must be the same as the profiles folder on the server.

Client customization

Besides the preferences that are exportable by KNIME Analytics Platform there are additional settings that can be added to the preference files to customize clients:

/instance/org.knime.workbench.explorer.view/defaultMountpoint/defaultMountpoints=<mount id1>,<mount id2>,…

A comma separated list of default mount points that should be loaded,e.g. `LOCAL,EXAMPLES,My-KNIME-Hub`. Changes to this list only affects new workspaces, i.e. workspaces which already contain default mount points will still contain them even though only they haven't been defined here. If this option is absent and `defaultMountpoint/enforceExclusion` isn't set to true then all default mount points will be added. The current default mount points are `LOCAL`, `EXAMPLES`, and `My-KNIME-Hub`.

/instance/org.knime.workbench.explorer.view/defaultMountpoint/enforceExclusion=<true|false>

If set to true then all default mount point not defined by `/instance/org.knime.workbench.explorer.view/defaultMountpoint/defaultMountpoints` will be removed on start up.

/instance/com.knime.customizations/helpContact.buttonText=<label, e.g. Contact Support>

If set together with `/instance/com.knime.customizations/helpContact.address` a button with the provided label will occur under `Help` in KNIME Analytics Platform. Clicking on the button will, depending on the `helpContact.address`, either open the default mail client or the default browser with the provided address.

/instance/com.knime.customizations/helpContact.address=<uri, e.g. mailto:support@company or https://company/support>

Sets the address of the support contact. This option only takes effect in combination with `/instance/com.knime.customizations/helpContact.buttonText`.

/instance/com.knime.customizations/documentation.buttonText=<label, e.g. Documentation>

Sets the label of the documentation button that can be found under `Help` in KNIME Analytics Platform. Clicking on the button will open the default browser and navigate to the documentation. If set to `-` the button will be hidden.

`/instance/com.knime.customizations/documentation.address=<uri, e.g. https://company/documentation or file:///sharedSpace/documentation>`

Sets the address of the documentation. By default the documentation address points to the KNIME documentation.

`/instance/com.knime.customizations/windowTitle.appendix=<appendix, e.g. sponsored by company>`

Adds the appendix to the window title of KNIME Analytics Platform.

`/instance/com.knime.customizations/updateSite.uris=<uri>,<uri>,…`

Adds the provided addresses to the update sites.

`/instance/com.knime.customizations/updateSite.names=<name>,<name>,…`

The names that are shown under `Available Software Sites` for the provided update sites of option. Note that the number of names must match the number of provided URIs.

`/instance/com.knime.customizations/updateSite.default.disable=<true|false>`

Disables the default added update sites added by KNIME after a fresh installation or update. If a user enables these update sites again they will remain enabled.

`/instance/com.knime.customizations/updateSite.default.forceDisable=<true|false>`

Disables the default added update sites added by KNIME after a fresh installation or update. If a user enables these update sites again they will be disabled with the restart of their client.

Security considerations

The following section describe some general security considerations for running a KNIME Server. Some of them are active by default, some other require manual configuration based on your specific environment.

Protecting configuration files

The configuration files must be accessible by the system account running the KNIME Server. However, this account also runs the KNIME Executor which executes the workflows. This means that a malicious workflow can in principle access the server configuration files if the absolute file system paths are known. Therefore, for high security environments we recommend removing write permissions on the configurations files from the system account so that at least the workflow cannot modify them. This includes the following directories and their contained files:

- `<tomee-folder>/conf`
- `<tomee-folder>/bin`
- `<tomee-folder>/endorsed`
- `<tomee-folder>/lib`
- `<server-repository>/config`

Encrypted communication

Communication between KNIME Analytics Platform and KNIME Server is performed via HTTP(S). By default, both unencrypted communication via HTTP and encrypted communication via HTTPS (SSL) is enabled.

All encryption is handled by TomEE, see the Tomcat SSL Configuration How-to for full documentation.

Server configuration

The KNIME Server installer will enable encryption using a generic server certificate that the client accepts. Note that most browsers will issue a certificate warning when you access the KNIME WebPortal via https for the first time. For production it is recommended to add your own certificate as follows:

1. Obtain a certificate and create a new Java keystore file named `knime-server.jks` as described in Tomcat SSL Configuration How-to
2. Replace the `<tomee-folder>/conf/knime-server.jks` with the keystore file created in the previous step (note: this will replace the generic server certificate)
3. Adjust the `certificateKeystorePassword` of the following "<Connector… />" definition found in `<tomee-folder>/conf/server.xml` to match the password used in the first step:

```
<Connector SSLEnabled="true" compression="on" maxThreads="150"
    protocol="org.apache.coyote.http11.Http11Nio2Protocol"
    port="8443" scheme="https" secure="true" server="Apache Tomcat">
    <SSLHostConfig protocols="TLSv1, TLSv1.1, TLSv1.2">
        <Certificate
            certificateKeystoreFile="conf/knime-server.jks"
            certificateKeystorePassword=<your password>
            type="RSA"/>
    </SSLHostConfig>
</Connector>
```

   You can also adjust the port number but you should not change any of the other value unless you understand the implications.

4. Restart TomEE.

If you want to enforce using only encrypted communications (HTTPS), we suggest to completely disable the unencrypted HTTP connector on port 8080 (by default). To do this remove the line that defines the first HTTP Connector in the `server.xml` or embed it into an XML comment so that it is not processed on startup.

Client configuration

If you want encrypted connection from KNIME Analytics Platform to KNIME Server, you have to make sure that KNIME accepts the server certificate. If you have a "real" certificate that was signed by a well-known certification authority then you should be save. If the signing CA is not known to Java you have to add the CA's certificate to the keystore used by KNIME:

1. Get the CA's certificate in PEM format.
2. Add the CA certificate to the JRE's keystore file in

```
`<knime-folder>/jre/lib/security/cacerts`
```

   (KNIME Analytics Platform 3.4.3 and older) or


```
`<knime-folder>/plugins/org.knime.binary.jre.<..>/jre/lib/security/cacerts`
```

(KNIME Analytics Platform 3.5.0 and newer). This is performed with the keytool command that is part of any Java installation (e.g. `<knime-folder>/<jre-folder>/bin/keytool`):

```
keytool -import -trustcacerts -alias <ca-alias> \
    -file <CA.crt> -keystore jre/lib/security/cacerts
```

You can choose an arbitrary name for `<ca-alias>`. For `<CA.crt>` insert your CA's certificate file. The password for the keystore file is "changeit".

Disabling the Manager application

The default KNIME Server installation does not add any users with permissions to access the manager application. The Tomcat manager application is not required for the correct functioning of KNIME Server. You may wish to disable the functionality by deleting the `manager`, `host-manager` and `ROOT` directories from your installation. Note that you should not delete the `ROOT` directory if you chose to install KNIME Server using the context root of `ROOT`.

Tomcat shutdown port

The Tomcat shutdown port is accessible on port 8005, which should not be accessible from machines other than localhost. We have renamed the `SHUTDOWN` command to a random string that is generated at installation time.

You may choose to remove this option completely by finding the following configuration in the `server.xml`:

```
<Server port="8005" shutdown="<RANDOMSTRING>">
```

and changing it to: <Server port="-1" shutdown="<RANDOMSTRING>">

CSRF prevention

Cross-site request forgery (CSRF) is a type of malicious exploit of a website where unauthorized commands are transmitted from a user that the website trusts (see the Wikipedia entry for more technical details). In the context of KNIME Server this means that

some other web page issues a (hidden) REST request to KNIME Server using the current user's active WebPortal session. The user usually doesn't notice anything but operations are performed with their account. Since version 4.3.4 KNIME Server contains a CSRF protection which prevents any modification requests (e.g. `POST`, `PUT`, or `DELETE`) to REST methods from hosts other than KNIME Server itself.

In case you have internal web pages on other hosts that deliberately perform valid requests you can disable CSRF protection by adding the following line to `<apache-tomee>/conf/Catalina/localhost/knime.xml`:

```
<Parameter name="com.knime.server.rest.csrf-protection" value="false"
    override="false" />
```

Avoid clickjacking attacks

Clickjacking is also a malicious attempt to trick a user into clicking on something different than perceived, potentially revealing confidential information or taking control of the computer. (See the Wikipedia entry for more technical details). The best option to avoid clickjacking is setting the HTTP header `X-Frame-Options` to an appropriate value to prevent the WebPortal being embedded in a third party website. In KNIME Server this can be done with a configuration option `com.knime.server.webportal.restrict_x_frame_options`. The value can be one of `DENY`, `SAMEORIGIN` or `ALLOW-FROM any_origin_url`. See also this article from MDN about more details of the header and available options.

Please note that, if you want to embed the WebPortal on a different website and want this setting to be enabled, you will have to set the value to `ALLOW-FROM xxx` (where `xxx` has to be replaced with the URL of the embedding website).

Hiding server details

By default, Tomcat prints its name and version number on error pages (e.g. if a location entered in the browser does not exist) and in standard HTTP headers. This information can be used by an attacker to target potential security issues for this particular version. Therefore for high security environments it's recommended to at least hide the server's version. Fresh installations from 4.5 onwards already hide the version. If you are upgrading from an existing installation, you can apply the following two small configuration changes:

- Add a file `<tomee-folder>/lib/org/apache/catalina/util/ServerInfo.properties` with the following contents:


```
server.info=Apache Tomcat
server.number=8.5.11.0
server.built= Jan 10 2017 21:02:52 UTC
```

Only the value of “server.info” is shown in error pages and by default includes the version number. The above example only exposes the server’s name.

- Modify the `<Connector>` entries in `<tomee-folder>/conf/server.xml` and add an attribute “server” with “Apache Tomcat” as value:

```
<Connector port="8080" *server="Apache Tomcat"* ... />
```

This change hides the server version in HTTP headers.

You may also choose to set the following parameter in the knime-server.config file. For full details see KNIME Server configuration file options:

com.knime.server.webportal.hide_version=true

Advanced settings

There are a couple more actions you can take to make the server and the application even more secure which we don’t discuss in detail here because they are only useful in special setups. Example are

- Cross-Origin Resource Sharing
- Strict Transport Security
- Content Security Policy (this policy cannot be implemented in Tomcat without writing custom code; see the section about Running behind frontend server for a possible solution)

Running behind frontend server

In some cases it makes sense to run KNIME Server (Tomcat) behind a frontend server. Examples are

- Running several KNIME Servers under the same (public) hostname
- Adding custom HTTP headers (e.g. Content Security Policy, see above)
- Reusing existing HTTPS configurations
- Using standard ports (80, 443)

No configuration changes are required on the KNIME Server side, however, the frontend server must ensure that

- the public hostname is passed to KNIME Server in all HTTP requests. See the example below for details, and
- information about the public protocol (HTTP or HTTPS) is passed onto the KNIME Server.

Otherwise links generated by KNIME Server may point to the internal address which is useless for outside clients and can even expose sensitive information. A sample configuration for Apache HTTPD looks as follows:

```
<VirtualHost *:443>
    ServerName public.knime.server

    # Make sure the public protocol is passed to the server;
    # not required if internal and external protocol are the same
    RequestHeader set X-Forwarded-Proto "https"

    # Ensure that the public hostname is also used in forwarded requests
    ProxyPreserveHost On
    ProxyRequests Off

    ProxyPass /tomee/ejb http://internal:8080/tomee/ejb
          keepalive=On nocanon
    ProxyPass /knime http://internal:8080/knime

    # Optional
    ProxyPass /com.knime.enterprise.sketcher.ketcher
          http://internal:8080/com.knime.enterprise.sketcher.ketcher
</VirtualHost>
```

Please note that such advanced setups require detailed knowledge about Tomcat and Apache configuration (or whatever frontend server you are using) and we can only provide

limited support.

KNIME WebPortal

The KNIME WebPortal requires workflow execution (see above for instructions). It can be accessed with any standard browser (see below for a list of supported browsers) at:

```
http://server-address:8080/knime/
```

If you enabled encrypted communication above, you can also use https:

```
https://server-address:8443/knime/
```

Supported browsers

The following browsers are supported by KNIME WebPortal, we do not actively test or support KNIME WebPortal in in older browser versions.

Google Chrome:	Version 70.0+
Internet Explorer	Version 11.0+
Microsoft Edge	Version 44.0+
Firefox Version	Version 63.0+ and 60 ESR
Safari	Version 12.0+

Most WebPortal functionality may work with older browsers, although this is not tested.

Please note that IE8 and below is not supported.

Customizing WebPortal layout

The layout and styles of the WebPortal are customizable and controllable using templates. The KNIME Server installation includes two example templates, `webportalTemplate.default`, and `webportalTemplate.goldMining`.

If you wish to change the look and feel, add your own components, or embed the WebPortal into your corporate environment you can use one of the templates in the `server-repository/config` folder in the server installation package for inspiration.

Gold mining WebPortal Customisation example login page

A custom template can be deployed under `<server repository>/config/webportalTemplate` (simply rename one of the example template folders as a starting point).

The customization can be done in 4 parts:

- The templates represent the structural elements of the different pages and panels of the WebPortal.
- A custom stylesheet can be applied by placing a CSS file called `knime_template.css` into the templates folder.
- In order to change behavior, add additional libraries or define JavaScript functions to be available for example to the JavaScript enabled nodes, place a JS file called `knime_template.js` into the templates folder.
- You can place a custom `favicon.ico` in the template folder in order to change the page/shortcut icon of all WebPortal pages.

Any additional resources (images, libraries, etc.) can be placed into this folder as well. On server startup, this folder is copied to a location that is accessible to a web browser. You can reference these resources directly in the CSS file (e.g. `background-image: url(logo.png);`) and prepend the custom folder in any template file (e.g. `<img src="custom/logo.png">`).

The template files are HTML fragments placed into the layout of the WebPortal page. There are several placeholders that are filled during runtime with the corresponding dynamic

components. A placeholder is defined by an HTML element that has a location attribute. The tables below list all possible placeholders.

To debug and troubleshoot your custom WebPortal templates please use the development tools of your preferred browser. A long-waiting request during page layout will most-likely be due to a mandatory component missing. Please also note that the CSS file is added to the page before the regular stylesheet. If you wish to overwrite styles you need to define new classes or ids in your template files or use the `!important` keyword.

Customizing the login page

The login page is customized in the file `knime_template_login.html` using the following placeholders:

Location ID	Mandatory	Placeholder for…
`knime-login-image`		The KNIME logo shown on the login page
`knime-version-label`		A label displaying the current server version
`knime-login-message`	Yes	Additional login message (e.g. session expired)
`knime-input-username`	Yes	The login input field for the username
`knime-input-password`	Yes	The login input field for the password
`knime-login-button`	Yes	The login button.

Customizing the main page

The main page is customized with several files. The overall structure is defined in `knime_template_main.html` and uses the following placeholders:

Location ID	Mandatory	Placeholder for…
`knime-header`	Yes	The header component

Location ID	Mandatory	Placeholder for…
`knime-main-panel`	Yes	The two sectioned main panel
`knime-footer`		The footer component

The main page's header and footer can further customized in `knime_template_header.html` or `knime_template_footer.html`, respectively. The following templates can be used in either file:

Location ID	Mandatory	Placeholder for…
`knime-user-label`		A label showing the current user name
`knime-version-label`		A label displaying the current server version
`current-year`		The current year

The following templates can be used only in the `knime_template_header.html` file:

Location ID	Mandatory	Placeholder for…
`knime-logo`		The KNIME logo
`knime-logout-button`	Yes	The logout button
`knime-admin-button`		A button to call the server administration panel. Only shown for admin users.
`knime-settings-button`		A button to call the current user's settings panel. Only shown if user management is available.

Customizing job pages

The start page of a job can be customized in `knime_template_job_start.html` using the

following templates:

Location ID	Mandatory	Placeholder for…
`workflow_name`		The workflow name
`workflow_description`		The additional workflow description
`workflow-variables-credentials`	Yes	The workflow variables and credentials panel
`workflow-notification-settings`	Yes	The workflow notification settings panel
`job-start-button`	Yes	The button to start a new workflow job

The panel shown during job execution can be customized in `knime_template_job_executing.html` with the following templates:

Location ID	Mandatory	Placeholder for…
`workflow_name`		The workflow name
`workflow_description`		The additional workflow description
`job-status`		A label displaying the current status of the job
`progress-indicator`	Yes	The spinning wheel progress indicator
`job-cancel-button`	Yes	A button to cancel execution

The panel showing intermediate steps during job execution (i.e. one quickform step) can be customized in `knime_template_job_prompt.html` with the following templates:

Location ID	Mandatory	Placeholder for…
`workflow_name`		The workflow name

Location ID	Mandatory	Placeholder for…
`workflow_description`		The additional workflow description
`knime-quickforms`	Yes	The panel displaying all available quickforms or JavaScript components
`job-buttons`	Yes	The button bar for back, next and discard buttons

The panel showing the final result page of a job can be customized in `knime_template_job_result.html` with the following templates:

Location ID	Mandatory	Placeholder for…
`workflow_name`		The workflow name
`workflow_description`		The additional workflow description
`job-success-text`		A label displaying if execution was successful
`knime-quickforms`	Yes	The panel displaying all available quickforms or JavaScript components
`job-buttons`	Yes	The button bar for back, next and discard buttons
`report-preview`	Yes	A component to display a preview of a report, if available
`job-warnings`	Yes	Component displaying all errors/warnings that occurred during execution

Customizing workflow group information

The page displaying information for a selected workflow group can be customized in `knime_template_directory.html` with the following templates:

Location ID	Mandatory	Placeholder for…
`workflow_name`		The workflow group name
`workflow_description`		The additional workflow group description

Installing a molecule sketcher

The KNIME WebPortal can be used with an integrated molecular sketcher. The server installation package contains additional WAR files in the `sketchers` folder. In order to use one of these sketcher copy the corresponding WAR file into `<tomee-folder>/webapps` and configure as described below. The WAR file should automatically be extracted into a folder of the same name. You may need to restart TomEE before the WAR is extracted.

Marvin JS sketcher

First check that you followed the instructions in the paragraph Installing a molecule sketcher. The current version of the Marvin Sketcher provided by Chemaxon is available at https://chemaxon.com/products/marvin-js/download. Download the sketcher code and extract its contents to

```
<tomee-folder>/webapps/com.knime.enterprise.sketcher.marvinJS/marvinJS/
```

Change the server configuration in `knime-server.config` and set

```
com.knime.server.webportal.sketcher_page=/com.knime.enterprise.sketcher.marvinJS/sketcher.html
```

Marvin sketcher applet (Deprecated)

You are most likely interested in the newer Marvin JS functionality, in which case check the previous section Marvin JS sketcher. If you're absolutely sure that this is the functionality that you're interested in, then please read on.

First check that you followed the instructions in the paragraph Installing a molecule sketcher. The current version of the Marvin Sketcher provided by Chemaxon is available at https://www.chemaxon.com/download/marvin/for-web-developers/. Download the sketcher applet and extract its contents to

```
<tomee-folder>/webapps/com.knime.enterprise.sketcher.marvin/marvin/
```

Change the server configuration in `knime-server.config` and set

```
com.knime.server.webportal.sketcher_page=/com.knime.enterprise.sketcher.marvin/sketcher.
html
```

GGA Ketcher

First check that you followed the instructions in the paragraph Installing a molecule sketcher. Change the server configuration in `knime-server.config` and set

```
com.knime.server.webportal.sketcher_page=/com.knime.enterprise.sketcher.ketcher/sketcher
.html
```

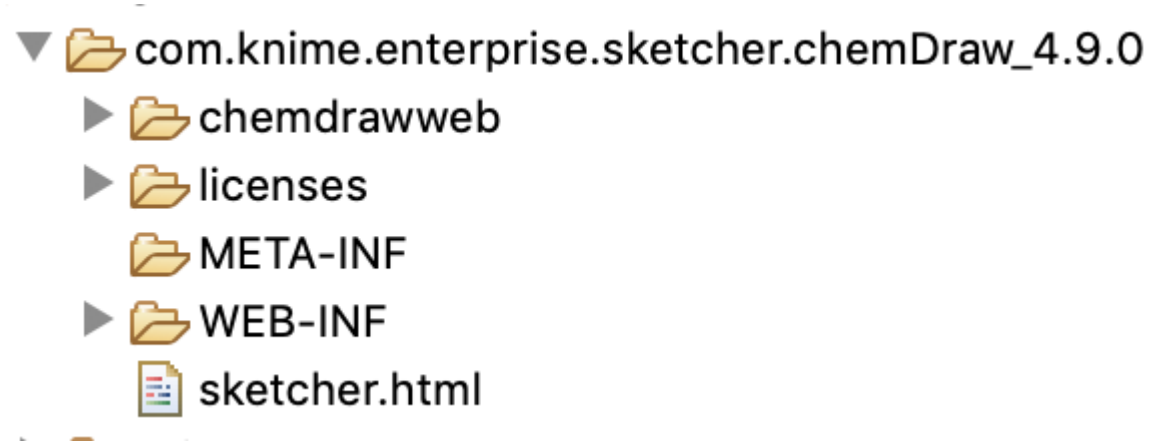
JSME Molecule Editor

First check that you followed the instructions in the paragraph Installing a molecule sketcher. Change the server configuration in `knime-server.config` and set

```
com.knime.server.webportal.sketcher_page=/com.knime.enterprise.sketcher.jsme/sketcher.ht
ml
```

ChemDraw JS Sketcher

First check that you followed the instructions in the paragraph Installing a molecule sketcher. Extract the package file containing the ChemDraw JS sketcher and locate the `chemdrawweb` folder. Copy this folder into the extracted web application for the sketcher integration in your TomEE `webapps` folder. Also copy your license file for ChemDraw JS in a `licenses` folder. Your folder structure should look similar to the following picture:



chemDraw JS Setup

Change the server configuration in `knime-server.config` and set

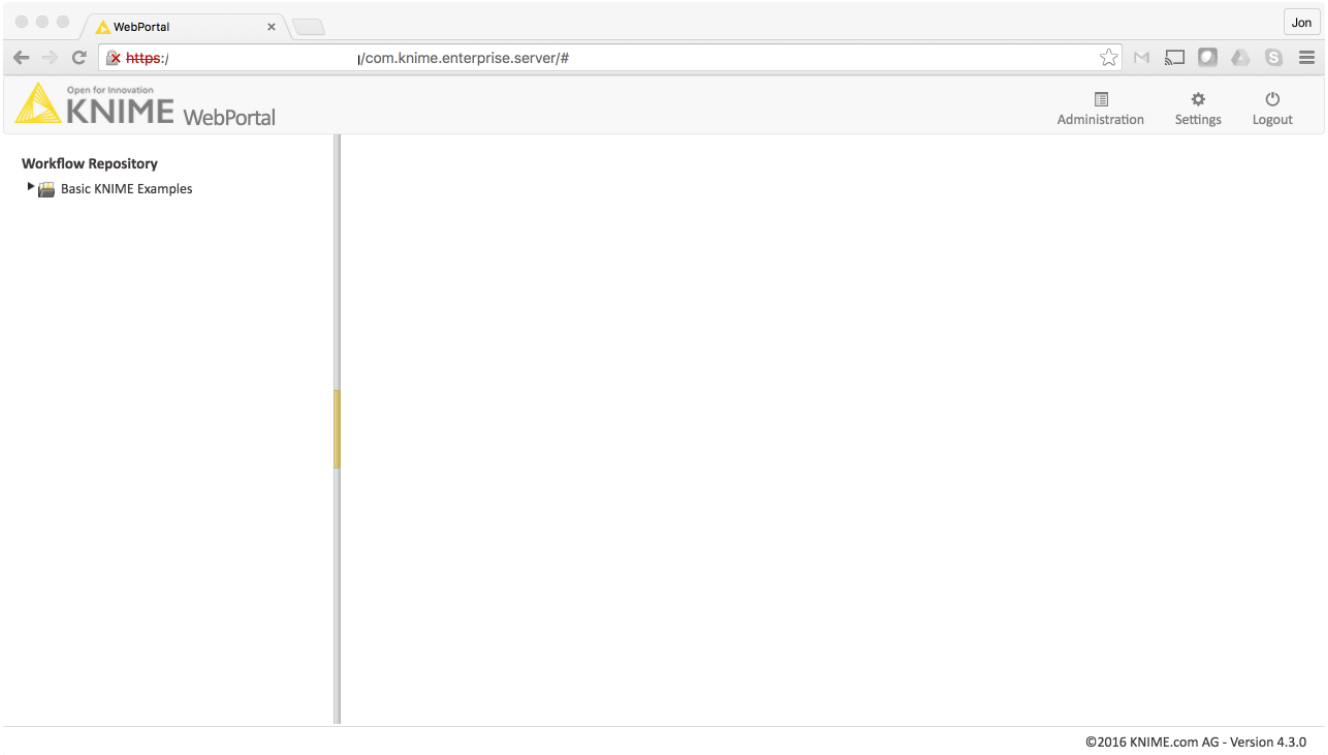
```
com.knime.server.webportal.sketcher_page=/com.knime.enterprise.sketcher.chemDraw/sketcher.html
```

Adapt the path if you have deployed the sketcher integration under a different context root.

Administration pages

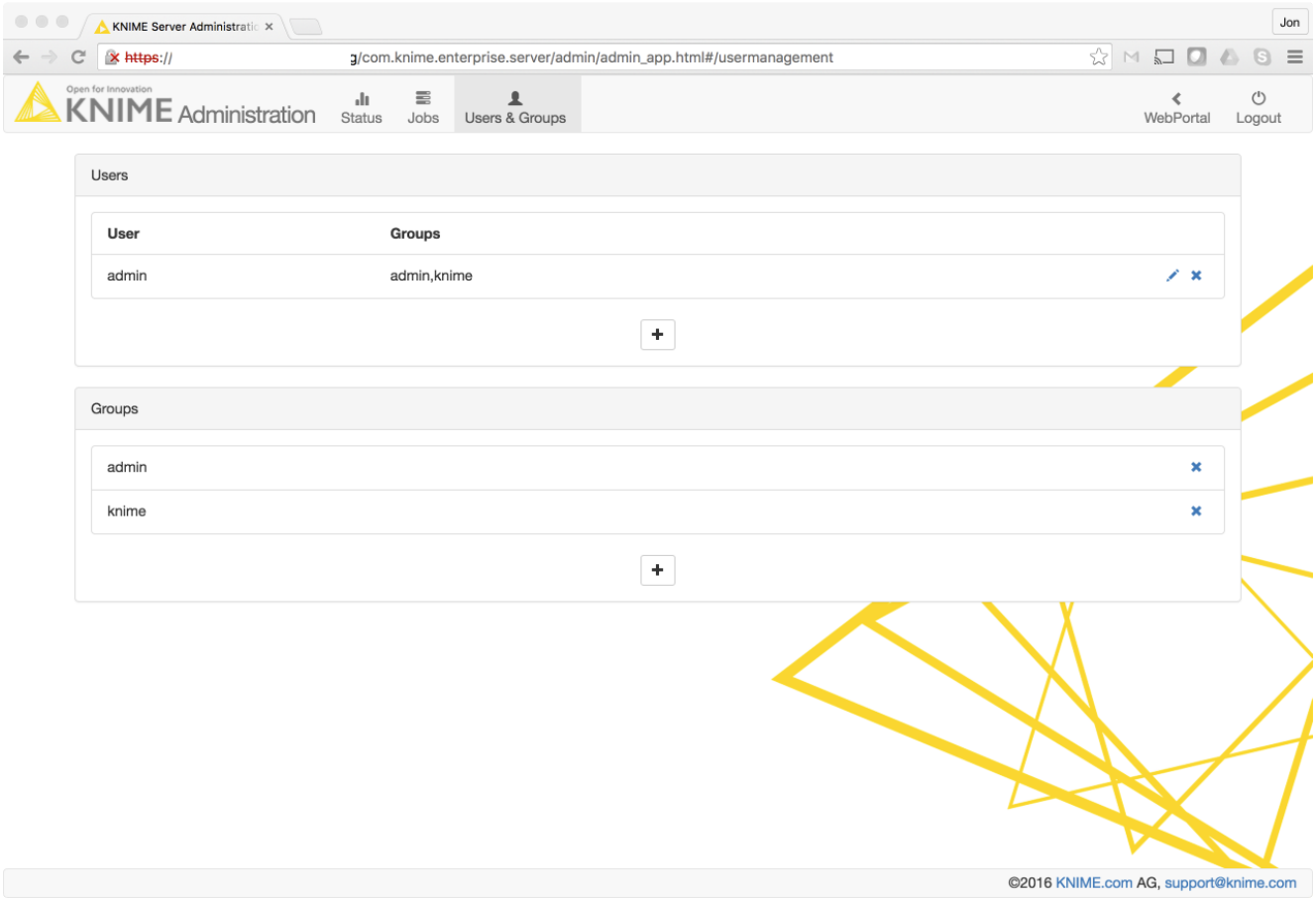
If the KNIME WebPortal is installed, a KNIME Server administration page will available, too. It can be accessed by logging into the WebPortal through a web browser and clicking on the corresponding link at the top right.

In order to have access to the administration page, a user must have administrator privileges. Users can be granted administrator privileges via the server configuration file (please see section Server administrator).



Login to the KNIME Server Admin pages by clicking on the Administration button on the top right.

The administration page provides details about the server's status (see Server status), an overview of workflow jobs (see Jobs), and allow to manage local users and groups (see User groups). Please note that the users and groups management will only be available with a specific server configuration as described in section User groups.



User and group administration page

Server status

The Server Status info box provides the following information:

Host	The name of the host the server is currently running on.
Version	The version of the running KNIME Server.
Uptime	The time that has been passed since the last start of the server.
Executors Info	Information about KNIME instances used to execute jobs. It includes the associated user name, the port a particular KNIME Executor is listening to, the uptime, general status and the number of jobs.

Config Info	Click on the item to expand it. It provides the current KNIME Server configuration as specified by the server's configuration file (see Section KNIME Server configuration file).

The License and Users Info-box provides information about the currently used license and the users. The following items are available:

License Type	Type of the used license.
Expiration Date	Expiration date of the license.
Company	The company the license has been issued for.
Host Identifiers	Host information used to check the license against. This can be, for instance, MAC- or IP-addresses.
Comment	An optional comment regarding the license.
Users	Click on the item to expand it. It displays the number of users and maximum number of allowed users for either the "Desktop", the "WebPortal", or the "Webservices". For WebPortal and Webservice the most recent login of users are provided as well.

Jobs

The jobs page lists all currently available jobs on KNIME Server. The table displays the job-status icon (move the mouse cursor over the icon for more details), its name, the workflow the job is associated with, its owner, and notifications. More details about a particular job can be accessed by clicking on the 'magnifier'-icon. For a more detailed description of the provided information for each job please refer to the documentation of the KNIME Server API version 4.x, `http://<server>:8080/com.knime.enterprise.server/rest/v4/\_profile/knime-server-doc-v4.xml` in particular the documentation of the resource `com.knime.enterprise.server.rest.api.v4.jobs.ent.WorkflowJob`.

By typing into the search field, only jobs that contain the search string (present in any of the

columns), will be shown.

Users & groups

Note: The users and groups management is only available if the database-based authentication is chosen (see section Database-based authentication). If the LDAP or file-based authentication is configured, the users and groups management will *not* be available.

Through the user management page, users can be added (the 'plus sign'), their groups and password changed (the pencil-icon) or deleted (the waste-bin-icon). The currently logged-in user cannot be deleted.

Furthermore, groups can be added or removed. If a group is removed all member will be removed from this group, too. The pre-configured "admin" group cannot be removed. To grant another user to access the administration pages, he must have administrator privileges (e.g. being member of an admin group).

Managing access to files/workflows/components

You can assign access permissions to each server item (workflows or workflow groups) to control the access of other users to your workflows and groups.

The owner

The server stores the owner of each server item, which is the user that created the item. When you upload a flow, copy a workflow, save a workflow job (an executed flow) or create a new workflow group you are assigned to the new item as owner. When a new server item is created, you can set the permissions how you want this item to be available to other users. Later on, only the owner can change permissions on an item.

User groups

When the KNIME Server administrator defines the users that have access to the KNIME Server, the users are assigned to groups. Groups can be defined as needed — for example one group per department, or per research group, etc. Each user must be in at least one group, and could be in many groups.
You can set a group to be an administrator group (with the configuration option “com.knime.server.server_admin_group=<group name>”). Users assigned to that group are considered server administrators.

Server administrator

Specific users can be set server administrator with a configuration option (`com.knime.server.server_admin_users=<user>,<user>,⋯,`) or by assigning them to the administrator group (see section User groups). Server administrators are not restricted by any access permissions. Administrators always have the right to perform any action usually controlled by user access rights. They can always change the owner of an item, change the permissions of an item, they see all workflow jobs (while regular users only see their own jobs) and they can delete all jobs and items.

Access rights

There are three different access rights that control access to a workflow and two for a workflow group:

Workflow group permissions

Read	Allows the user to see the content of the workflow group. All workflows and subgroups contained are shown in the repository view.
Write	If granted, the user can create new items in this workflow group. He can create new subgroups and can store new workflows or Shared Components in the group. Also deletion of the group is permitted.

Note: In order to access a workflow it is not necessary to have read-permissions in the workflow group the flow is contained in. Only the listing of contained flows is controlled by the read-right. Also, a flow can be deleted without write permission in a group (if the corresponding permission on the flow is granted).

Also, in order to add a flow to a certain group, you only need permissions to write to that particular group, not to any parent group.

Workflow permissions

Execute	Allows the user to execute the flow, to create a workflow job from it. It does not include the right to download that job, or even store the job after it finishes (storing requires the right to download).
Write	If granted, the user can overwrite and delete the workflow.
Read	Allows the user to download the workflow (including all data stored in the flow) to its local desktop repository and inspect the flow freely.

Note: Executing or downloading/reading a flow does not require the right to read in the group that contains the flow. In fact, there is currently no right controlling the visibility of a single flow (there is no "hidden" attribute).

Access to workflow jobs and scheduled jobs

There are no permissions to be set on a workflow job or a scheduled job. Only the owner — the user that created the job — can see the job in the repository view, and he is the only user that can delete it (besides any server administrator).

In order to store a workflow job as new workflow in the server's repository, the user needs the right to download the original workflow (the flow the job was created from). (This is a requirement, because the newly created workflow is owned by the user that stores the job — and the owner can easily grant itself the right to download the flow. Thus, if the original flow didn't have the download right set, the user that is allowed to execute the flow could easily work around the missing download right.)

"Owner", "Group", and "Other" rights

As the owner of a server item (workflow, shared component or workflow group) you can grant access rights to other users. But you can only assign permissions on a group level, not for particular users.

Owner rights

The owner can assign permissions to himself to protect a flow from accidental deletion. He can change his own permissions at any time.

Group rights

The owner of a server item can assign permissions to all users of a specific group. If an access right is granted to a group, all users that are assigned to this group have this right.

"Other" rights

Permissions can be set to all users that are not the owner and that are not in one of the groups.

Note: Access rights are adding up and can't be withdrawn — that means, if, for example, you grant the right to execute a flow to "other" users and you define permissions for a certain group of users not including the execute right, these users of that group are still able to execute that flow, as they obtain that right through the "other" permissions.

Webservice interfaces

RESTful webservice interface

KNIME Server supports execution of workflows via a REST interface. The entry point for the REST interface is `http://server-address/knime/rest/`.

The interface is based on a hypermedia-aware JSON format called Mason. Details about the interface, its operations, endpoints and message formats are provided at the following locations (best opened in an internet browser):

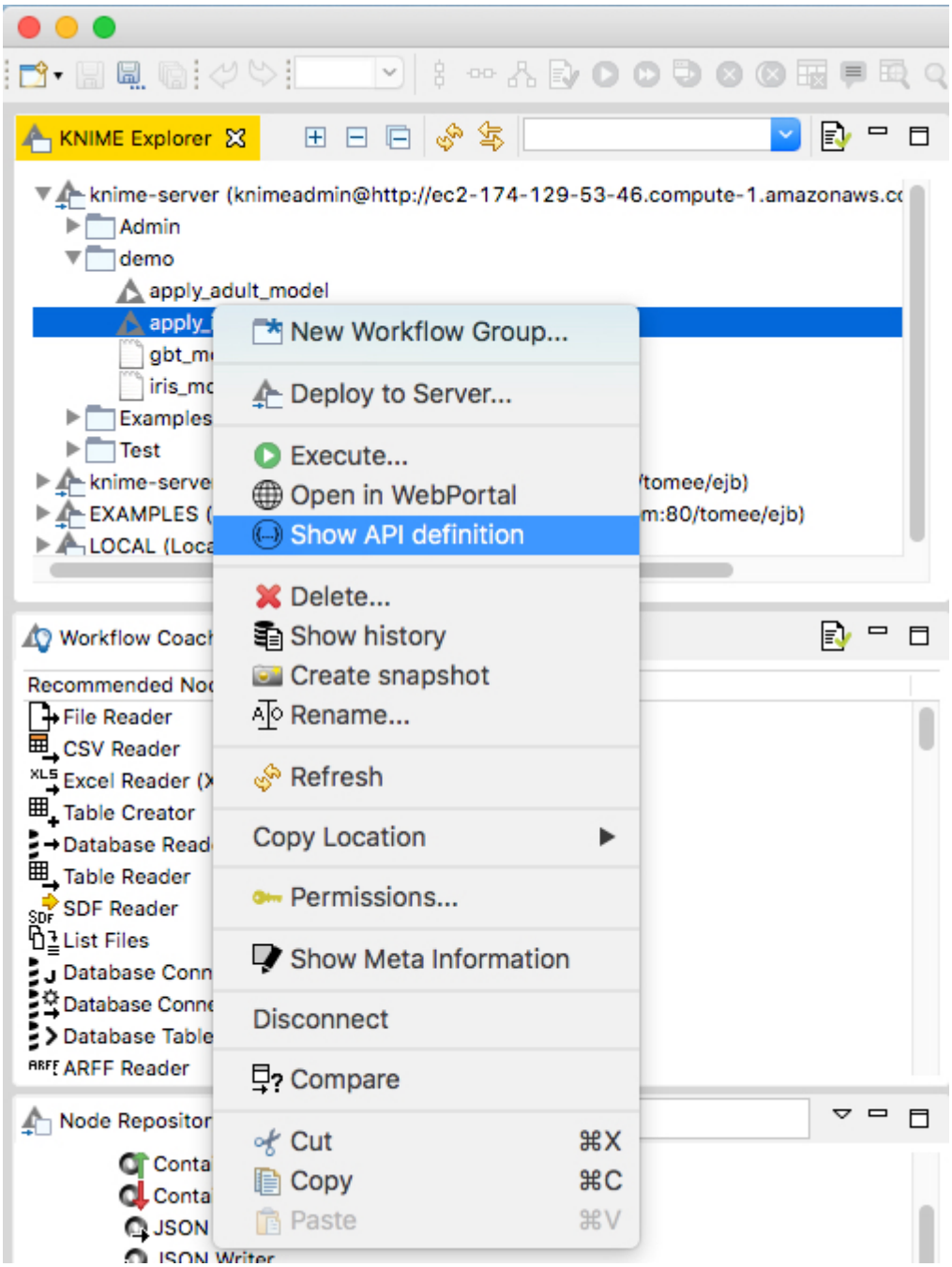
- `http://<server-address>/knime/rest/_profile/knime-server-doc.xml` for the general interface and
- `http://<server-address>/knime/rest/v4/_profile/knime-server-doc-v4.xml` for the 4.x API

(see also the "Link" HTTP header in all responses returned by the server).

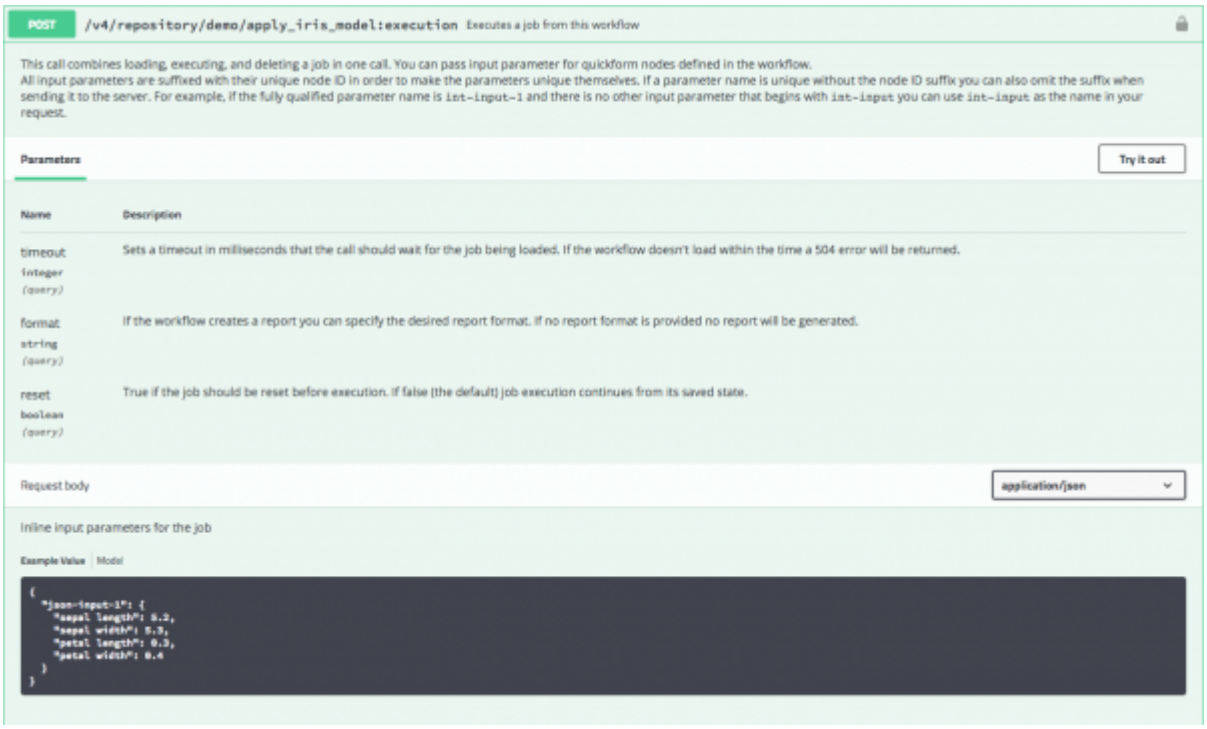
The usual starting point to query the repository and to execute operations is
`http://<server-address>/knime/rest/v4/repository/`
(note the trailing '/'). The returned document also contains links to further operations.

SwaggerUI for Workflows

The KNIME Server automatically generates SwaggerUI pages for all workflows that are present on the KNIME Server. From the KNIME Analytics Platform you can access that functionality using the `Show API definition` context menu item.



Clicking the menu item will open a SwaggerUI page for that workflow in your browser. It's also possible to browse to that page using the REST API as described in the above section.



Common problems

Always reset with flow variables

If the values of flow variables are changed in the remote execution dialog, the flow must be reset in order for the new values to be propagated. In this case, don't remove the checkmark "Reset before Execution" in the execution dialog.

knime.ini file not found

If the KNIME instance that is used to execute flows on the server doesn't seem to have the settings specified in the `knime.ini` file, it is possible that the server didn't find the ini-file: The server takes the default ini-file from the same folder as the KNIME executable. If you specify a wrapper script as executable that is located outside the installation folder it doesn't find the default ini-file. In this case copy the ini-file from the installation folder into `<server-repository>/config`.

Server startup takes a long time

In some cases it may take quite some time (up to several minutes) until the server responds to requests on Linux systems.

Insufficient entropy

This is usually caused by insufficient entropy for the random number generator used by Tomcat. You can work around this issue by specifying a different random number source, which will provide numbers faster but which are also less random:

1. Edit `<tomee-folder>/conf/catalina.properties`.
2. Add a line `java.security.egd=file:/dev/./urandom` at the bottom of the file (note the "/./")
3. Restart TomEE

Large number of jobs

In cases where the KNIME Server retains a large number of jobs then it may be necessary to increase the amount of memory that TomEE can access. Simply edit the file setenv.bat

(Windows) or setenv.sh (Linux) to increase the value of -Xmx to double the current setting.

Changelog (KNIME Server 4.10)

KNIME Server 4.10.6 (released November 8, 2021)

Bugfixes

- [SRV-3612] - Directory path traversal when requesting client profiles[1]
- [WEBP-876] - Cross-Site-Scripting vulnerability in old WebPortal login[2]

KNIME Server 4.10.5 (released November 4, 2020)

Enhancements

- [SRV-3123] - RabbitMQ HA support

KNIME Server 4.10.4 (released August 26, 2020)

Enhancements

- [SRV-2580] - Allow re-connecting to existing jobs after server restart
- [SRV-2712] - Executor should retry connections to message queue
- [SRV-2974] - Add hidden setting to use localhost instead of 127.0.0.1 as Oauth callback URL
- [SRV-2979] - Suppress warnings about missing metainformation for simple files during workflow group upload
- [SRV-3064] - Improve Explorer perfomance for many jobs
- [SRV-3110] - Increase maximum number of in-memory jobs in local queue-based executor

Bugfixes

- [SRV-2934] - Wrong error message when loading workflow failed in executor
- [SRV-2942] - Edit mount point dialog doesn’t show address to REST path
- [SRV-2943] - server_logs.zip sometimes contains folders with incomplete executor ID
- [SRV-2965] - Stackoverflow in Qpid breaks message queue connections

- [SRV-2978] - Regular job status updates may prevent job swapping in distributed executors
- [SRV-3025] - Account settings of KNIME server are not applied on runtime
- [SRV-3046] - ETag when fetching the repository item is null
- [SRV-3052] - Server sends multiple Status Emails for loops in workflows
- [SRV-3054] - No different font for disabled schedules visible via REST connection to the server
- [SRV-3069] - Potential deadlock when modifying scheduled job on old workflows
- [SRV-3117] - Reading files from the server repository randomly fails with 403

KNIME Server 4.10.3 (released April 6, 2020)

Enhancements

- [SRV-2705] - Remove info message about node recommendation update checkup

Bugfixes

- [SRV-2910] - Scheduled Jobs not persisted fully when created with EJB
- [SRV-2918] - Starting KNIME Server with wrong secret will delete all schedules
- [SRV-2718] - Closing KNIME OAuth Pop-Up (ESC Key) Will Block Port Used For Redirect
- [SRV-2895] - Workflows cannot be copied when scheduled job from other user exists
- [SRV-2901] - Long node messages lead to communication errors between RMI executor and server
- [SRV-2917] - Server installer doesn't set JAVA_HOME properly in automated installations

KNIME Server 4.10.2 (released March 5, 2020)

Enhancements

- [SRV-2750] - Improve execute dialog in case workflow configuration cannot be loaded

Bugfixes

- [SRV-2579] - Job requirements are ignored when restoring swapped job
- [SRV-2787] - Edit Mount Point dialog resizes if password is entered
- [SRV-2851] - Nondeterministic behavior when setting input for job in job-pool
- [SRV-2859] - Executing a pooled job with two identical parameter names leads to duplicate key error.
- [SRV-2854] - An updated scheduled job fails to skip execution if an older job is still running
- [SRV-2865] - KNIME Explorer hangs when trying to disconnect from the server after it was just restarted
- [SRV-2879] - NullPointerExceptions from executor during swapping are not handled in server
- [SRV-2742] - Certain wizard execution jobs cannot be found via REST any more
- [SRV-2875] - EJB explorer cannot connect to server when scheduled job exists

KNIME Server 4.10.1 (released February 5, 2020)

Enhancements

- [SRV-2433] - Allow updating linked components from EXAMPLES on KNIME Hub
- [SRV-2729] - Allow blacklisting of nodes to prevent execution on server
- [SRV-2789] - Remove checkbox from job configuration dialog
- [SRV-2806] - [OAuth] Show message when trying to login to KNIME Server/Hub via OAuth and server is unreachable
- [SRV-2856] - Encrypt workflow configuration for scheduled jobs
- [SRV-2796] - Improve *Deploy to server* dialog

Bugfixes

- [SRV-2857] - Java 8u242 breaks execution with RMI executors
- [SRV-2836] - Remote Workflow Editor: Ignore server version qualifier when comparing versions
- [SRV-2845] - Cannot rename components via EJB explorer

- [SRV-1463] - Server installer fails when used with Java 9+
- [SRV-2751] - Load timeout while retrieving job configuration may lead to orphaned job in executor
- [SRV-2766] - [OAuth] Server Restart While Being Logged in via AP Using Basic Auth Leads To Unexpected Redirect
- [SRV-2803] - [OAuth] Logging in Through AP With User Not Present in Tomcat Realm Spams Log With "Cannot determine roles for…"
- [SRV-2804] - Log Parser and Usage Reporting workflow fails when no uploads are present in log
- [SRV-2842] - RMI update script does not update reporting extension
- [SRV-2852] - Scheduled job does not save password of credentials configurations
- [SRV-2855] - Permissions panel hides first line on macOS

KNIME Server 4.10.0 (released December 6, 2019)

Enhancements

- [SRV-1828] - Support for multiple update sites in Management: Client Preferences/Personalisation
- [SRV-2037] - Core license file for individual executors
- [SRV-2172] - Action on Call Workflow Dialog for job execution dialog
- [SRV-2224] - Create primitive dialog to edit job configuration inputs
- [SRV-2225] - Create advanced dialog to edit job configuration inputs via gateway API
- [SRV-2274] - Examples: Add REST workflow to swap owners to installer
- [SRV-2382] - Improved error message in Explorer when no server license available
- [SRV-2432] - Configure retries for KNIME Server to attempt to connect to queue
- [SRV-2518] - License order of precedence
- [SRV-2570] - Frontend: Setup WebPortal API layer
- [SRV-2571] - Frontend: repository tree prototype
- [SRV-2572] - Frontend: workflow page prototype
- [SRV-2573] - Frontend: job page prototype
- [SRV-2587] - Fetch Identity Provider Endpoints on Login
- [SRV-2588] - Provide Identity Provider Endpoints Before Login When Using OAuth

- [SRV-2594] - Add OIDC Authentication to KNIME Server
- [SRV-2595] - Add Redirect Page For KNIME Server Authentication
- [SRV-2610] - Additional meta information for wizard page
- [SRV-2629] - Load Pagebuilder from executor
- [SRV-2648] - Add workflow configuration for Call Workflow action
- [SRV-2649] - Set actions on job creation
- [SRV-2666] - Infer 'user friendly' job status from job-response
- [SRV-2687] - Add http cache headers to gateway web-resource responses
- [SRV-2709] - Server report endpoint to serve content 'inline'
- [SRV-2716] - Turn off executor rotation as default
- [SRV-2719] - Add documentation to the Admin Guide on how to customize the node repository
- [SRV-2727] - Add embedded queue to knime-server.config

Bugfixes

- [SRV-2371] - Reset problem in remote workflow editor
- [SRV-2479] - Gateway API: wizard page endpoints return escaped string instead of json
- [SRV-2633] - Workflow cannot be overriden when scheduled job of another user exists
- [SRV-2658] - Applying new preferences requires restarting twice on Windows
- [SRV-2694] - mail.smtp.from parameter not set by installer
- [SRV-2701] - ETags should include authenticated username for filtered resources
- [SRV-2728] - Local File Chooser throws a nullpointer exception on execute dialog
- [SRV-2737] - Failure to load workflow may kill executor
- [SRV-2753] - Fix wrong color inherit on user-icon in header in edge
- [SRV-2772] - Local groups with spaces cannot be deleted on KNIME Server Small

Third party software licenses

The KNIME Server software makes use of third-party software modules, that are each licensed under their own license. Some of the licenses require us to note the following:

The following libraries are used and licensed under the CDDL v1.1 and are owned by Oracle. The copyright belongs to the respective owners.

- javax.json-1.0.4.jar
- javax.json-api-1.0.jar
- jstl-1.2.jar

The following libraries are used and licensed under the Apache 2.0 license. The copyright belongs to the respective owners.

- amqp-client-5.5.0.jar
- animal-sniffer-annotations-1.14.jar
- bcel-5.2.jar
- bson4jackson-2.9.2.jar
- commons-compress-1.15.jar
- commons-fileupload-1.3.1.jar
- commons-io-2.4.jar
- error_prone_annotations-2.0.18.jar
- guava-23.0.jar
- httpclient-4.5.3.jar
- httpcore-4.4.6.jar
- j2objc-annotations-1.1.jar
- jackson-annotations-2.8.0.jar
- jackson-core-2.8.11.jar
- jackson-databind-2.8.11.jar
- jackson-dataformat-xml-2.8.11.jar
- jackson-datatype-jdk8-2.8.11.jar
- jackson-datatype-jsr310-2.8.11.jar

- jackson-datatype-jsr353-2.8.11.jar
- jackson-module-jaxb-annotations-2.8.11.jar
- javassist-3.21.0-GA.jar
- je-7.4.5.jar
- jsr305-1.3.9.jar
- keycloak-tomcat-adapter-7.0.0.jar
- objenesis-2.6.jar
- ognl-3.0.8.jar
- org.osgi.compendium-4.3.1.jar
- org.osgi.core-4.3.1.jar
- qpid-bdbstore-7.0.6.jar
- qpid-broker-core-7.0.6.jar
- qpid-broker-plugins-amqp-0-8-protocol-7.0.6.jar
- rmiio-2.1.0.jar
- stax-api-1.0.1.jar
- stax2-api-3.1.4.jar
- thymeleaf-2.1.4.RELEASE.jar
- txtmark-0.13.jar
- unbescape-1.1.0.RELEASE.jar
- vaadin-client-compiled-7.7.9.jar
- vaadin-server-7.7.9.jar
- vaadin-shared-7.7.9.jar
- vaadin-themes-7.7.9.jar
- woodstox-core-5.0.3.jar
- xmlbeans-2.5.0.jar

The following libraries are used and licensed under the MIT license. The copyright belongs to the respective owners.

- jsoup-1.8.3.jar
- slf4j-api-1.7.25.jar

- jquery 2.2.4
- lodash 4.17.4
- react-15.6.2
- react-bootstrap 0.29.5
- react-bootstrap-table 3.3.4
- react-dom 15.6.2
- react-sidebar 2.1.1

The following libraries are used and licensed under the BSD 3-clause license. The copyright belongs to the respective owners.

- Node-forge 0.7.4 (Copyright (c) 2010, Digital Bazaar, Inc. All rights reserved.)

The following libraries are used and licensed under the Do what the fuck you want to public license. The copyright belongs to the respective owners.

- `reflections-0.9.10.jar`

CDDL v1.1

1. Definitions. 1.1. "Contributor" means each individual or entity that creates or contributes to the creation of Modifications. 1.2. "Contributor Version" means the combination of the Original Software, prior Modifications used by a Contributor (if any), and the Modifications made by that particular Contributor. 1.3. "Covered Software" means (a) the Original Software, or (b) Modifications, or (c) the combination of files containing Original Software with files containing Modifications, in each case including portions thereof. 1.4. "Executable" means the Covered Software in any form other than Source Code. 1.5. "Initial Developer" means the individual or entity that first makes Original Software available under this License. 1.6. "Larger Work" means a work which combines Covered Software or portions thereof with code not governed by the terms of this License. 1.7. "License" means this document. 1.8. "Licensable" means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein. 1.9. "Modifications" means the Source Code and Executable form of any of the following: A. Any file that results from an addition to, deletion from or modification of the contents of a file containing Original Software or previous Modifications; B. Any new file that contains any part of the Original Software or previous Modification; or C. Any new file that is contributed or otherwise made available under the terms of this License. 1.10. "Original Software" means the Source Code and Executable form of computer software code that is originally released under this License. 1.11. "Patent Claims" means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent

Licensable by grantor. 1.12. "Source Code" means (a) the common form of computer software code in which modifications are made and (b) associated documentation included in or with such code. 1.13. "You" (or "Your") means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License. For legal entities, "You" includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, "control" means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity. 2. License Grants. 2.1. The Initial Developer Grant. Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, the Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license: (a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer, to use, reproduce, modify, display, perform, sublicense and distribute the Original Software (or portions thereof), with or without Modifications, and/or as part of a Larger Work; and (b) under Patent Claims infringed by the making, using or selling of Original Software, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Software (or portions thereof). (c) The licenses granted in Sections 2.1(a) and (b) are effective on the date Initial Developer first distributes or otherwise makes the Original Software available to a third party under the terms of this License. (d) Notwithstanding Section 2.1(b) above, no patent license is granted: (1) for code that You delete from the Original Software, or (2) for infringements caused by: (i) the modification of the Original Software, or (ii) the combination of the Original Software with other software or devices. 2.2. Contributor Grant. Conditioned upon Your compliance with Section 3.1 below and subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license: (a) under intellectual property rights (other than patent or trademark) Licensable by Contributor to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof), either on an unmodified basis, with other Modifications, as Covered Software and/or as part of a Larger Work; and (b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: (1) Modifications made by that Contributor (or portions thereof); and (2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination). (c) The licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first distributes or otherwise makes the Modifications available to a third party. (d) Notwithstanding Section 2.2(b) above, no patent license is granted: (1) for any code that Contributor has deleted from the Contributor Version; (2) for infringements caused by: (i) third party modifications of Contributor Version, or (ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or (3) under Patent Claims infringed by Covered Software in the absence of Modifications made by that Contributor. 3. Distribution

Obligations. 3.1. Availability of Source Code. Any Covered Software that You distribute or otherwise make available in Executable form must also be made available in Source Code form and that Source Code form must be distributed only under the terms of this License. You must include a copy of this License with every copy of the Source Code form of the Covered Software You distribute or otherwise make available. You must inform recipients of any such Covered Software in Executable form as to how they can obtain such Covered Software in Source Code form in a reasonable manner on or through a medium customarily used for software exchange. 3.2. Modifications. The Modifications that You create or to which You contribute are governed by the terms of this License. You represent that You believe Your Modifications are Your original creation(s) and/or You have sufficient rights to grant the rights conveyed by this License. 3.3. Required Notices. You must include a notice in each of Your Modifications that identifies You as the Contributor of the Modification. You may not remove or alter any copyright, patent or trademark notices contained within the Covered Software, or any notices of licensing or any descriptive text giving attribution to any Contributor or the Initial Developer. 3.4. Application of Additional Terms. You may not offer or impose any terms on any Covered Software in Source Code form that alters or restricts the applicable version of this License or the recipients' rights hereunder. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Software. However, you may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear that any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer. 3.5. Distribution of Executable Versions. You may distribute the Executable form of the Covered Software under the terms of this License or under the terms of a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable form does not attempt to limit or alter the recipient's rights in the Source Code form from the rights set forth in this License. If You distribute the Covered Software in Executable form under a different license, You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer. 3.6. Larger Works. You may create a Larger Work by combining Covered Software with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Software. 4. Versions of the License. 4.1. New Versions. Oracle is the initial license steward and may publish revised and/or new versions of this License from time to time. Each version will be given a distinguishing version number. Except as provided in Section 4.3, no one other than the license steward has the right to modify this License. 4.2. Effect of New Versions. You may always continue to use, distribute or otherwise make the

Covered Software available under the terms of the version of the License under which You originally received the Covered Software. If the Initial Developer includes a notice in the Original Software prohibiting it from being distributed or otherwise made available under any subsequent version of the License, You must distribute and make the Covered Software available under the terms of the version of the License under which You originally received the Covered Software. Otherwise, You may also choose to use, distribute or otherwise make the Covered Software available under the terms of any subsequent version of the License published by the license steward. 4.3. Modified Versions. When You are an Initial Developer and You want to create a new license for Your Original Software, You may create and use a modified version of this License if You: (a) rename the license and remove any references to the name of the license steward (except to note that the license differs from this License); and (b) otherwise make it clear that the license contains terms which differ from this License. 5. DISCLAIMER OF WARRANTY. COVERED SOFTWARE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED SOFTWARE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED SOFTWARE IS WITH YOU. SHOULD ANY COVERED SOFTWARE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED SOFTWARE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER. 6. TERMINATION. 6.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive. 6.2. If You assert a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You assert such claim is referred to as "Participant") alleging that the Participant Software (meaning the Contributor Version where the Participant is a Contributor or the Original Software where the Participant is the Initial Developer) directly or indirectly infringes any patent, then any and all rights granted directly or indirectly to You by such Participant, the Initial Developer (if the Initial Developer is not the Participant) and all Contributors under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively and automatically at the expiration of such 60 day notice period, unless if within such 60 day period You withdraw Your claim with respect to the Participant Software against such Participant either unilaterally or pursuant to a written agreement with Participant. 6.3. If You assert a patent infringement claim against Participant alleging that the Participant Software directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into

account in determining the amount or value of any payment or license. 6.4. In the event of termination under Sections 6.1 or 6.2 above, all end user licenses that have been validly granted by You or any distributor hereunder prior to termination (excluding licenses granted to You by any distributor) shall survive termination. 7. LIMITATION OF LIABILITY. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED SOFTWARE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU. 8. U.S. GOVERNMENT END USERS. The Covered Software is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" (as that term is defined at 48 C.F.R. § 252.227-7014(a)(1)) and "commercial computer software documentation" as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Software with only those rights set forth herein. This U.S. Government Rights clause is in lieu of, and supersedes, any other FAR, DFAR, or other clause or provision that addresses Government rights in computer software under this License. 9. MISCELLANEOUS. This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by the law of the jurisdiction specified in a notice contained within the Original Software (except to the extent applicable law, if any, provides otherwise), excluding such jurisdiction's conflict-of-law provisions. Any litigation relating to this License shall be subject to the jurisdiction of the courts located in the jurisdiction and venue specified in a notice contained within the Original Software, with the losing party responsible for costs, including, without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License. You agree that You alone are responsible for compliance with the United States export administration regulations (and the export control laws and regulation of any other countries) when You use, distribute or otherwise make available any Covered Software. 10. RESPONSIBILITY FOR CLAIMS. As between Initial Developer and the Contributors, each party is responsible for claims and damages arising,

directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability. NOTICE PURSUANT TO SECTION 9 OF THE COMMON DEVELOPMENT AND DISTRIBUTION LICENSE (CDDL) The code released under the CDDL shall be governed by the laws of the State of California (excluding conflict-of-law provisions). Any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California and the state courts of the State of California, with venue lying in Santa Clara County, California. The GNU General Public License (GPL) Version 2, June 1991 Copyright (C) 1989, 1991 Free Software Foundation, Inc. 51 Franklin Street, Fifth Floor Boston, MA 02110-1335 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. Preamble The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too. When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things. To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it. For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights. We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software. Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations. Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all. The precise terms and conditions for copying, distribution and modification follow. TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION 0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the

terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you". Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does. 1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program. You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee. 2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions: a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change. b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License. c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it. Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program. In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not

bring the other work under the scope of this License. 3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following: a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.) The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable. If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code. 4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance. 5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it. 6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License. 7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the

conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program. If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances. It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice. This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License. 8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License. 9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation. 10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally. NO WARRANTY 11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.

SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. 12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. END OF TERMS AND CONDITIONS How to Apply These Terms to Your New Programs If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms. To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found. One line to give the program's name and a brief idea of what it does. Copyright (C) <year> <name of author> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1335 USA Also add information on how to contact you by electronic and paper mail. If the program is interactive, make it output a short notice like this when it starts in an interactive mode: Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details. The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program. You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names: Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker. signature of Ty Coon, 1 April 1989 Ty Coon, President of Vice This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License. Certain

source files distributed by Oracle America, Inc. and/or its affiliates are subject to the following clarification and special exception to the GPLv2, based on the GNU Project exception for its Classpath libraries, known as the GNU Classpath Exception, but only where Oracle has expressly included in the particular source file's header the words "Oracle designates this particular file as subject to the "Classpath" exception as provided by Oracle in the LICENSE file that accompanied this code." You should also note that Oracle includes multiple, independent programs in this software package. Some of those programs are provided under licenses deemed incompatible with the GPLv2 by the Free Software Foundation and others. For example, the package includes programs licensed under the Apache License, Version 2.0. Such programs are licensed to you under their original licenses. Oracle facilitates your further distribution of this package by adding the Classpath Exception to the necessary parts of its GPLv2 code, which permits you to use that code in combination with other independent modules not licensed under the GPLv2. However, note that this would not permit you to commingle code under an incompatible license with Oracle's GPLv2 licensed code by, for example, cutting and pasting such code into a file also containing Oracle's GPLv2 licensed code and then distributing the result. Additionally, if you were to remove the Classpath Exception from any of the files to which it applies and distribute the result, you would likely be required to license some or all of the other code in that distribution under the GPLv2 as well, and since the GPLv2 is incompatible with the license terms of some items included in the distribution by Oracle, removing the Classpath Exception could therefore effectively compromise your ability to further distribute the package. Proceed with caution and we recommend that you obtain the advice of a lawyer skilled in open source matters before removing the Classpath Exception or making modifications to this package which may subsequently be redistributed and/or involve the use of third party software. CLASSPATH EXCEPTION Linking this library statically or dynamically with other modules is making a combined work based on this library. Thus, the terms and conditions of the GNU General Public License version 2 cover the whole combination. As a special exception, the copyright holders of this library give you permission to link this library with independent modules to produce an executable, regardless of the license terms of these independent modules, and to copy and distribute the resulting executable under terms of your choice, provided that you also meet, for each linked independent module, the terms and conditions of the license of that module. An independent module is a module which is not derived from or based on this library. If you modify this library, you may extend this exception to your version of the library, but you are not obligated to do so. If you do not wish to do so, delete this exception statement from your version.

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and

improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

You must give any other recipients of the Work or Derivative Works a copy of this License; and

You must cause any modified files to carry prominent notices stating that You changed the files; and

You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such

third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support,

warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS+

MIT License

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE

New BSD License (3-clause)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of <company name> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

[1] The server managed customizations functionality of KNIME Server up to version 4.13.3 is vulnerable to Directory Path Traversal attacks. By manipulating variables that reference files by prepending “dot-dot-slash (../)” sequences and their variations or by using absolute file paths, it is possible to access arbitrary files and directories stored on the file system including application source code, configuration, and database. Due to the file-based architecture of KNIME Server, this vulnerability allows stealing users' data such as password hashes, workflows, licenses, jobs, and so on. No authentication is required to exploit this vulnerability. The vulnerability was found and reported by Dawid Czarnecki from NATO and is recorded as CVE-2021-44726

[2] The old KNIME WebPortal login page up to version 4.13.3 contains a DOM-based XSS vulnerability that once exploited, can be used to run any action as a victim user via malicious JavaScript. If the victim user is an administrator, it could be used to create a new administrator. To exploit the vulnerability it is required to create a specially crafted URL and convince the victim to open it. No authentication is required to exploit the vulnerability, however, authenticated users can be targeted. The vulnerability was found and reported by Dawid Czarnecki from NATO and is recorded as CVE-2021-44725

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