



PEGA MARKLOGIC CONNECTOR

DECEMBER 2022



Contents

- GENERAL INFORMATION 1**
 - Component Description..... 1
 - Compatibility..... 1
- INSTALLATION 1**
 - Installation of the MarkLogic ECM Services..... 1
 - Installation of the Pega MarkLogic Connector 3
- CONFIGURATION 4**
 - Configure the Repository Instance..... 4
 - Configure Metadata and Text Extraction..... 6
- UPDATE PEGA TO USE THE REPOSITORY 7**
- FEATURE LIST 8**
- SMART SHAPES 9**
 - The component adds a new Automation Smart Shape to workflow editing:..... 9
- RELEASE NOTES 10**

GENERAL INFORMATION

COMPONENT DESCRIPTION

The Pega MarkLogic Connector implements the Pega Repository API for the MarkLogic NoSQL Database.

MarkLogic is an Operational and Transactional Enterprise NoSQL Database that is multi-model, natively storing data as JSON, XML and RDF. Clients that want to use Pega Infinity and MarkLogic can now have case attachments and content be stored in a MarkLogic database.

By adding this component to a Pega application, you can integrate your application with a MarkLogic database using the new Repository interface. You can select MarkLogic as a Repository type when creating a new Repository. By enabling the use of the MarkLogic repository from the Application definition, all attachments and content will automatically be stored in the MarkLogic database.

More on Repository API is available here: <https://community.pega.com/knowledgebase/articles/custom-repository-types>.

COMPATIBILITY

Pega 8.5.3 or greater

INSTALLATION

INSTALLATION OF THE MARKLOGIC ECM SERVICES

In order to use the Pega MarkLogic Component, you need an existing MarkLogic cluster that has been configured with the corresponding backend Enterprise Case Management (ECM) services to support the Pega MarkLogic Component.

The connector distribution can be downloaded from developer.marklogic.com. It contains a Gradle project that will configure MarkLogic to expose the required REST endpoints and transforms.

In order to deploy this project's application, you'll first need the following software installed:

- Java 8 to 16 (required by Gradle)
- MarkLogic 10.0 (preferably the latest version)

The project uses Gradle but it includes [the Gradle wrapper](#), which means you don't need to install Gradle locally.

The project also uses the [Gradle properties plugin](#) to select the corresponding properties file to use to set the host, username, and password for deploying to an instance of MarkLogic.

To deploy the project to MarkLogic, first unzip the distribution archive that you downloaded from developer.marklogic.com.

First Time Installation

In the "marklogic-pega-connector-1.1.x" folder, create a gradle properties file for your environment. For example, for the "local" environment, create `./gradle-local.properties` and define the following properties:

- mlHost ("localhost" if you're running MarkLogic locally; otherwise, the host of your remote MarkLogic cluster)
- mlUsername (the username to run the deployment as)
- mlPassword (the password for the user defined by mlUsername)

Use the mlDeploy task to deploy the services. If no "env" is specified, it defaults to "local" (the "-i" flag is included for info-level logging, which can be helpful for debugging if there is an error).

```
./gradlew -i mlDeploy
```

If you want to deploy to an environment other than "local", for example "test", create `./gradle-test.properties` and define the properties as listed above and use the following to deploy:

```
./gradlew -i mlDeploy -Penv=test
```

Note: For Windows users, replace `./gradlew` with `gradlew.bat` above.

Enabling HTTPS

The default installation of the ECM services will configure MarkLogic to use HTTP. To configure the backend services to use HTTPS, follow the "[Configuring SSL on App Servers](#)" instructions in the MarkLogic documentation. Note that if you do not configure the server with a certificate from a well-known CA, you will need to configure Pega with a custom keystore (see below).

Once this has been run, you can install the Pega MarkLogic Connector and configure it to point to your instance of MarkLogic, as described below in the "Installation of the Pega MarkLogic Connector" section.

Upgrading

If the ECM services have been previously deployed to MarkLogic, upgrading to new services requires combining the existing configuration with the new package.

Copy your existing gradle properties file for your environment to the new "marklogic-pega-connector-1.1." folder. For example, for the "local" environment, copy `./gradle-local.properties` to the new folder.

If any other configuration files have been changed or added in the previous connector package, copy those files into the new "marklogic-pega-connector-1.1.x" folder (e.g. a customized app server configuration).

If upgrading an environment that has had HTTPS enable for all app servers, add the following properties to the `/gradle-test.properties`

- mlSsl=true
- systemProp.java.net.ssl.trustStore=<trust store with your CA>
- systemProp.java.net.ssl.trustStorePassword=<your trust store password>

Install the new service code and update the version info in the configuration by running

```
./gradlew -i mlLoadModules upgradeConfig
```

If upgrading an environment other than "local", for example "test", specify the environment

```
./gradlew -i mlLoadModules upgradeConfig -Penv=test
```

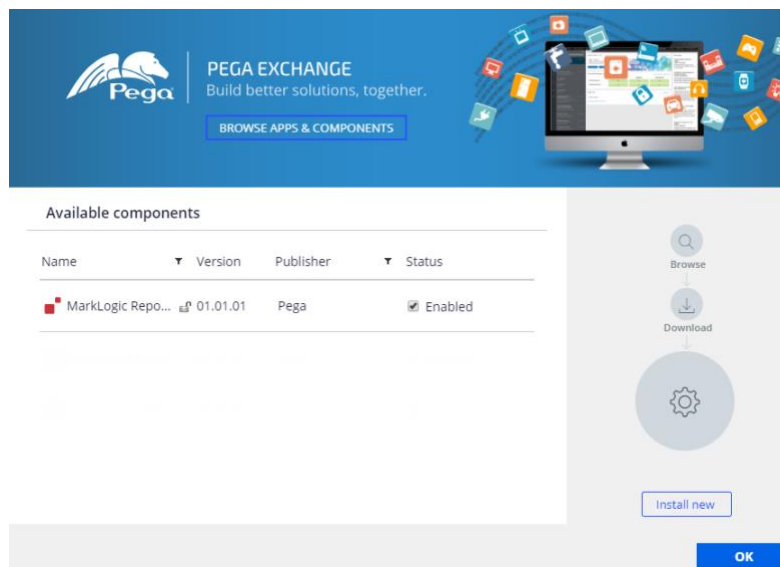
Note: For Windows users, replace “./gradlew” with “gradlew.bat” above.

INSTALLATION OF THE PEGA MARKLOGIC CONNECTOR

The Pega MarkLogic Repository Component is contained in the `marklogic-peg-connector-1.1.x/rulesets/MarkLogicRepository_1.1.x.zip` file that was created when the distribution archive was unzipped above.

First Time Installation

When install for the first time, open the Application rule form then, under the Enabled components section, click on Manage components button and then click on Install new and browse to the already downloaded component and then click OK.



Verify that the Component is listed on the Application rule form and save the Application rule.

Note: A server restart is required after importing the component into Pega.

Upgrading

When upgrading a previously installed version of the component, you must use the Import Wizard. Navigate to the wizard by selecting Import under the Application > Distribution sub menus from the Configure dropdown in Dev Studio. Select the `marklogic-peg-connector-1.1.x/rulesets/MarkLogicRepository_1.1.x.zip` file and follow the steps in the wizard. More information is available on the Import Wizard in [Pega documentation](#).

Note: A server restart may be required after upgrading the component.

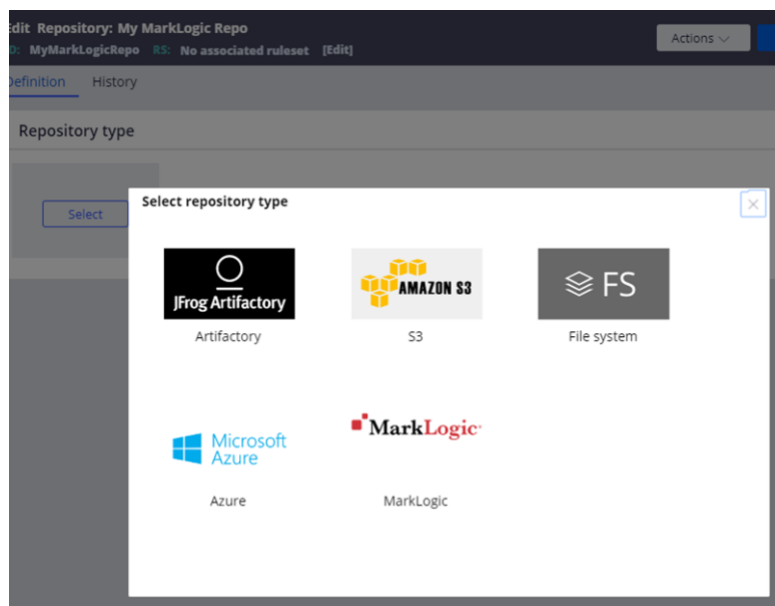
CONFIGURATION

The Pega MarkLogic Repository Component provides an implementation of the Repository API.

Important: Prior to configuring the Repository instance, make sure the Operator account that will be creating the Repository has the PegaRULES:RepositoryAdministrator role.

CONFIGURE THE REPOSITORY INSTANCE

- 1) Create a New Repository rule (Create->SysAdmin->Repository), provide your Repository Name/Description.
- 2) On the Repository Type selection screen, you will have a MarkLogic repository. Select it:



3. Fill in the Repository Settings:

Edit Repository: MarkLogic
ID: MarkLogic RS: No associated ruleset [Edit] [Delete] [Actions] [Save]

Definition History

Repository type

MarkLogic

Change

Repository configuration

Host name: [1]

Keystore Name: [2]

Configuration

Authentication profile: [3]

Root path: [4]

Path for Cases JSON: [5]

Collections [6]

Case Collection:

Case History Collection:

Attachments Collection:

Folder Collection:

Case JSON Transform [7]

transform - Option for Case JSON:

Search [8]

options - Search Option for File/Folcer Filter:

Advanced [9]

Create Folder structure for Case Attachments

Use UUID Storage Design. Attachments stored independent of Case

Test connectivity

Validate repository

Fields

[1] **Host Name:** URL with the host name and port for a MarkLogic cluster that has been configured with the required backend services for the connector. This should be a complete URL (including the port if needed) that indicates whether HTTP (http://) or HTTPS (https://) should be used.

[2] **Keystore Name:** If using HTTPS, provide the name of the Pega keystore rule containing the certificate. You must ensure that the keystore is added to the platform according to the “[Managing X.509 certificates](#)” Pega documentation. If HTTPS is not used or an additional certificate is not needed, leave this value blank.

[3] **Authentication Profile:** Create a new Basic Authentication Profile with the credentials for the MarkLogic account. Make sure 'Pre-emptive authentication' is checked.

[4] **Root Path:** Root URI for Pega documents. Used to Test Connectivity

[5] **Path for Cases JSON:** Root URI for where Case JSON documents are placed

[6] **Collections:** The different types of documents from Pega will be tagged by the collections string are defined here.

- Case Collection: Used for Case JSON documents
- Attachment Collection: File attachments to case/pulse
- Folder Collection: Folder objects

[7] **Case JSON Transform:** The REST document transform options for Case JSON documents.

[8] **Search Options:** The pre-configured search option that filters for Attachment/Folder collections.

[9] **Advanced:** Toggle on to create dedicated Case URI paths.

- Example: `/pega/case/B-1/hello123_B-1.txt` rather than `/pega/case/hello123_B-1.txt`

CONFIGURE METADATA AND TEXT EXTRACTION

In addition to the settings exposed via the Pega repository configuration screen, there are settings that can be changed on the backend that control how metadata and text are extracted from binary attachments.

- **extractMetadata** – Set to true if the metadata extraction, including extraction of any text contained in binary files, should be run (defaults to true)
- **extractMetadataMaxFileSize** – The maximum file size for files that will have their metadata extracted (defaults to 10 MB)
- **extractMetadataUnknownMimeTypes** – If set to true, metadata will be extracted even if the extension of the file is an unknown mime type (defaults to false)

A MarkLogic administrator can configure these by manually updating the `/pega/config.json` in the database. An example of a complete configuration document is shown below.


```
{
  "connectorVersion": "1.1.x",
  "caseCollection": "pega-case",
  "attachmentCollection": "pega-attachment",
  "attachmentMetadataCollection": "pega-attachment-metadata",
  "folderCollection": "pega-folder",
  "extractMetadata": true,
  "extractMetadataMaxFileSize": 10485760,
  "extractMetadataUnknownMimeTypes": false
}
```

UPDATE PEGA TO USE THE REPOSITORY

Update the Application Integration & Security tab to toggle on the Content Storage from the new Repository:

Content management

Configure the storage location for case and pulse attachments. Optionally configure sour

 **Content storage**

Store in Pega database
 Store in CMIS system
 Store in web storage provider
 Store in repository

Repository: Target folder:

Prompt users for confirmation before deleting content from repository

Browse the Repository to select the root path where all the Pega content will be stored:

Select folder

- /pega/attachment
- /pega
- /pega/case

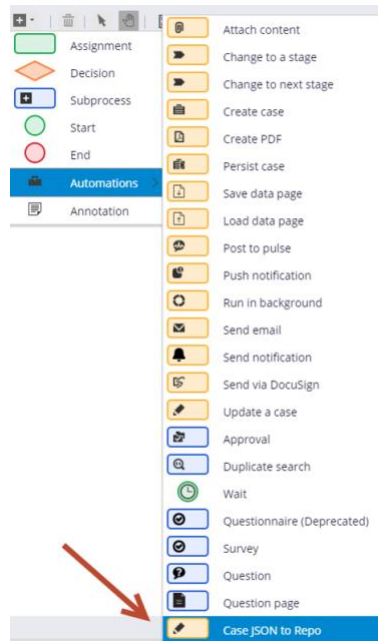
FEATURE LIST

Included Data Pages

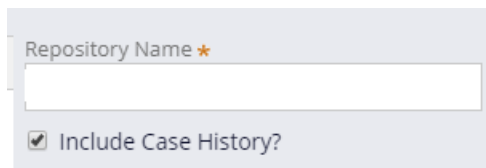
Page Name	MarkLogic Endpoint [Methods]	Description
D_MLRepoBasicSearch	v1/search [GET]	Enables Basic Search with options, collection, directory, q, view and pageLength parameters.
D_MLRepoDelete	v1/search [DELETE] v1/documents [DELETE] v1/resources/files [DELETE]	Repository API page. Depending on the advanced toggles, will delete/recursively delete based on path/uri.
D_MLRepoExists	v1/documents [GET] v1/resources/files [GET]	Repository API page. Depending on the advanced toggles, will check for the existence of a file/document based on path/uri.
D_MLRepoGetFile	v1/documents [GET] v1/resources/files [GET]	Repository API page. Depending on the advanced toggles, will get the file/document based on path/uri. Note: Using this page will convert streams into strings and is used for testing only.
D_MLRepolsAuthenticated	v1/documents [GET]	Repository API page. Will ping the repository to check if the AuthenticationProfile is valid.
D_MLRepoListFiles	v1/search [GET]	Repository API page. Will query the repository to get back the files/folders based on an initial folder uri.
D_MLRepoNewCase	v1/documents [PUT]	Will push a JSON document representing a case to the repository.
D_MLRepoNewCaseHistory	v1/documents [PUT]	Will push a JSON document representing a case history to the repository.
D_MLRepoNewFile	v1/documents [PUT] v1/resources/files [PUT]	Repository API page. Will create/push a document to the repository for a specific uri/path. Note: Using this page will convert streams into strings and is used for testing only.
D_MLRepoNewFolder	v1/documents [PUT]	Repository API page. Will create/push a folder "document" to the repository for a specific uri/path.

SMART SHAPES

The component adds a new Automation Smart Shape to workflow editing:



The Smart Shape takes the following parameters:

A screenshot of the configuration form for the 'Case JSON to Repo' Smart Shape. It features a text input field labeled 'Repository Name' with an asterisk indicating it is required. Below the input field is a checked checkbox labeled 'Include Case History?'.

- **Repository Name:** Name of the Repository Rule
- **Include Case History flag:** To toggle sending the case history

The Smart Shape will call D_MLRepoNewCase and D_MLRepoNewCaseHistory if the flag is set. It will use the current case context when sending the JSON data.

RELEASE NOTES

1.0.0 – November 04, 2021

- Initial release

1.0.1 – January 31, 2022

- Updated Log4J to address security vulnerability CVE-2021-44832

1.1.0 – April 04, 2022

- Added support for HTTPS
- Added streaming support to handle large binaries
- Patched for Spring4Shell vulnerability
- Miscellaneous documentation updates

1.1.1 – April 29, 2022

- Removed unused parameter from ruleset

1.1.2 – August 10, 2022

- Fixed memory leak in the Pega ruleset when uploading images
- Enhanced log messages in the Pega rulesets
- Improved documentation about how to upgrade

1.1.3 – October 28, 2022

- Fixed the check for duplicates when the “Create folder structure for case attachments” setting is not enabled

1.1.4 – December 5, 2022

- Fix for MIME base64 encoding issue

About MarkLogic

For over a decade, organizations around the world have come to rely on MarkLogic to power their innovative information applications. As the world's experts at integrating data from silos, MarkLogic's operational and transactional Enterprise NoSQL database platform empowers our customers to build next generation applications on a unified, 360-degree view of their data. Headquartered in Silicon Valley, MarkLogic has offices throughout the U.S., Europe, Asia, and Australia.

© 2022 MARKLOGIC CORPORATION. ALL RIGHTS RESERVED. This technology is protected by U.S. Patent No. 7,127,469B2, U.S. Patent No. 7,171,404B2, U.S. Patent No. 7,756,858 B2, and U.S. Patent No 7,962,474 B2. MarkLogic is a trademark or registered trademark of MarkLogic Corporation in the United States and/or other countries. All other trademarks mentioned are the property of their respective owners.



333 TWIN DOLPHIN DRIVE, SUITE 380, REDWOOD CITY, CA 94065
+1 650 655 2300 · +1 877 992 8885 · www.marklogic.com · sales@marklogic.com