Business Rule Maintenance Web Application

Documentation

Version 0.3

Content

Overview

The Business Rules Maintenance web application allows to create, update and delete projects, rulegroups, subgroup, rules and actions. These are used to define and orchestrate a logic with the purpose to execute it against a set of data.

The web application exports a project to a zip file which can directly be executed with JaRE – Java Rule Engine.

The ruleengine JaRE is written in Java and can run either standalone from the command line or may be embedded in other Java related projects. It may be embedded in other tools or web applications.

The main goal of using a business rules and ruleengine approach is to separate the IT logic from the business logic. This way both types of logic can be managed individually and by the relevant domain experts: IT expert or business expert. This creates a proper division of responsibilities, makes IT code cleaner and easier to maintain and thus adds to the overall quality and agility of the system.

By separating IT and business logic, the business user is not confronted with business rules being mixed with complex IT code or work flows. This enhances the transparency for the user.

This document explains the installation, configuration, concepts and usage of the web application.

Ruleengine Use Cases

The ruleengine can be used to:

- Filter (remove, route) data based on rule engine results
- Manipulate/update data based on rule engine results
- Calculate values based on rule engine results
- Create detailed logs of the results for further processing
- Create KPI calculations from data
- Quality checks of data

Prerequisites

To use the web application following prerequisites have to be met:

- Java: The Business Rules maintenance web application is written in Java. It comes in the form of a .war file (web archive).
- Apache Tomcat: Apache Tomcat allows to run Java code on the server side. Any other tool with similar capabilities and allowing to use .war files may also be used.
- MySQL/MariaDB: The web application stores configuration and all data entered by the user in a MySQL database. Other dialects of MySQL such as MariaDB may also be used.
- A moderately modern web browser. Parts of the application use Javascript and JQuery to enhance the user experience.

It is assumed that the user – before running the Business Rules Maintenance web application - has a running instance of Java, Tomcat and MySQL already in place.

The web application uses the JaRE Java library (a jar file). In case you want to use/install a newer version of this library see the section “Software Updates“.

Installation

1. MySQL/MariaDB: database setup

The first step is to download the schema of the database. The schema contains the definitions of the required database tables and some base configuration data.

Make sure that the MySQL or MariaDB server is actually running. Import the database schema into the running MySQL/MariaDB instance by issuing following command on the command line:

```
mysql -u [user] -p < ruleengine_rules.sql
```

Replace "[user]" in the command with an existing user that has sufficient rights to import the database schema and create the database, tables, indexes and data.

If your MySQL/MariaDB server does not require a username and password then leave out the apprropriate parameters in the command.

2. Apache Tomcat: install the web application

Make sure that Tomcat is actually running. Download the Business Rules Maintenance web application .war file.

Locate where in your system Apache Tomcat is installed. It contains a folder labeled "*webapps*". Copy the .war file into the "*webapps*" folder. Usually Tomcat automatically expands the war file (installs it) and the web application is immediately available. If this is not the case you might have to restart Tomcat.

Open a web browser window and enter following URL (address):

http://localhost:[port]/rule_maintenance/

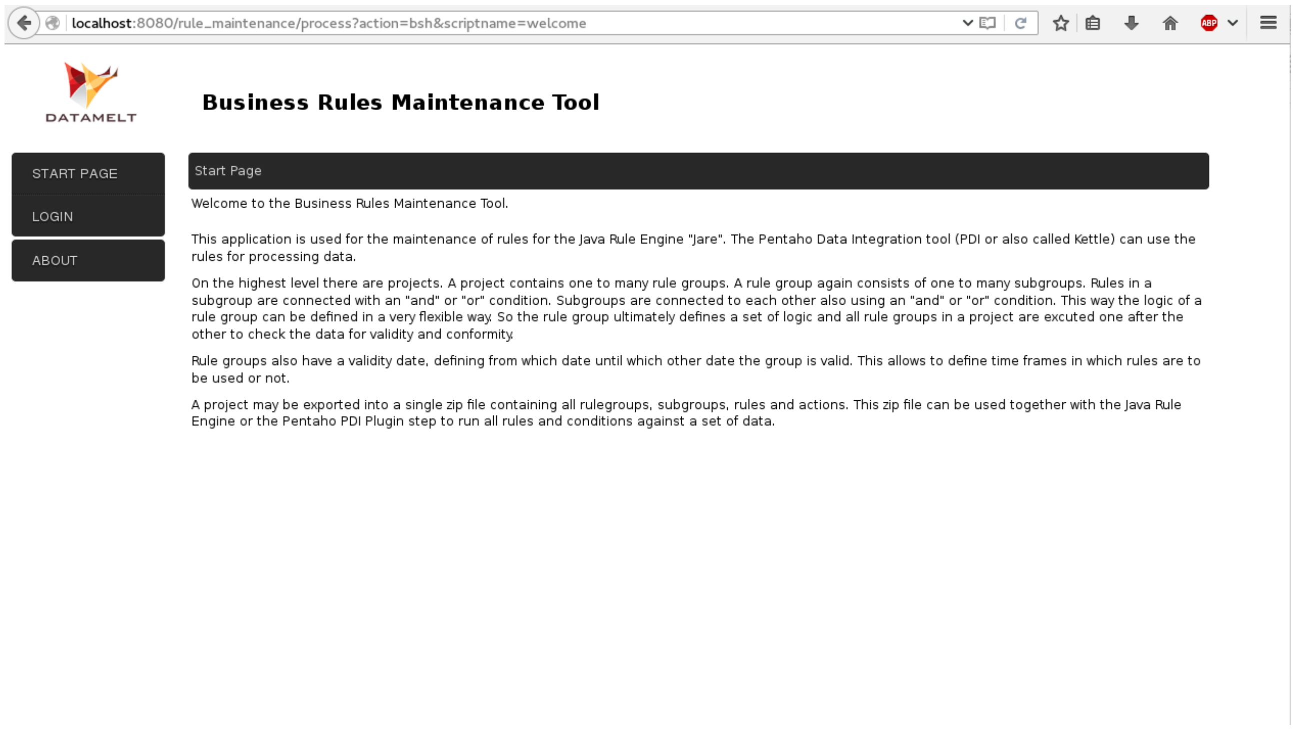
If Tomcat is running on a different server (not localhost), replace "*localhost*" with the server ip address or the hostname for that server. Tomcat usually runs on port 8080. If unsure, check with your Tomcat configuration, which port is used.

You should now see a page of the Business Rule Maintenance web application. On first start-up you may be presented with a configuration page. Enter the details for the database configuration: the host of the MySQL/MariaDB server and the port, the name of the database (the default is: "ruleengine_rules"), the user to access the database and the user password. Finally click on "save" to save the configuration.

Next you will be presented with the login page. If not, click on "Login" in the menu on the left-hand side. Now enter the userid and password for the web application. The default userid is *admin* and the password is also *admin*.

Start Page

Below is a screenshot and overview of the web application Start Page.


localhost:8080/rule_maintenance/process?action=bsh&scriptname=welcome
DATAMELT
Business Rules Maintenance Tool
START PAGE
LOGIN
ABOUT
Start Page

Welcome to the Business Rules Maintenance Tool.

This application is used for the maintenance of rules for the Java Rule Engine "Jare". The Pentaho Data Integration tool (PDI or also called Kettle) can use the rules for processing data.

On the highest level there are projects. A project contains one to many rule groups. A rule group again consists of one to many subgroups. Rules in a subgroup are connected with an "and" or "or" condition. Subgroups are connected to each other also using an "and" or "or" condition. This way the logic of a rule group can be defined in a very flexible way. So the rule group ultimately defines a set of logic and all rule groups in a project are excuted one after the other to check the data for validity and conformity.

Rule groups also have a validity date, defining from which date until which other date the group is valid. This allows to define time frames in which rules are to be used or not.

A project may be exported into a single zip file containing all rulegroups, subgroups, rules and actions. This zip file can be used together with the Java Rule Engine or the Pentaho PDI Plugin step to run all rules and conditions against a set of data.

Screenshot 1: Application Start Page

Menu

The menu is located on the left side of the web application and is always visible. It allows the user to navigate through the application.

The actual menu items displayed will vary depending on if the user is logged in or not and if the user is assigned to the "*Admin*" group – some of the menu items are only available to admin users.

Below is a description of the individual menu items.

Start Page

The Start Page is the first page a user is directed to after starting the web application or logging into the application. It shows a basic introduction to the purpose of the application.

Password

Allows to change the password of the user. The user has to provide the current password and then twice a new password.

If the application is configured to use LDAP, then a change of the password from within the application is not possible.

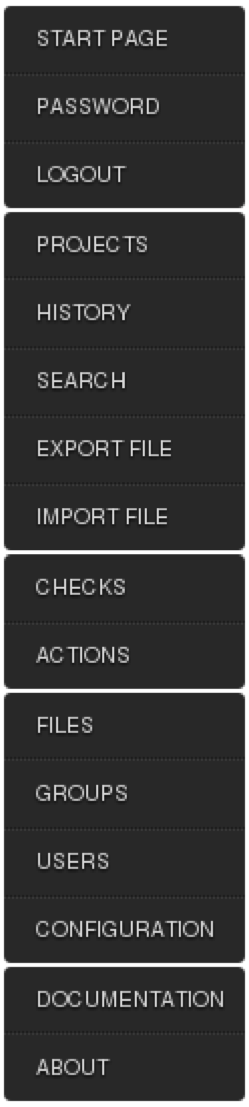
Login/Logout

The web application requires users to enter a userid and password to authenticate. On login it is determined to which group(s) the user is assigned. The user settings steer to which project users have access or which project they see.

Projects

Projects is the central place to work with projects, to create, update or delete rulegroups, subgroups, rules and actions.

Project is the highest hierarchy. A project may contain zero, one or multiple rulegroups. Rulegroups may contain zero, one or multiple subgroups. Rulegroups also may contain zero, one or multiple actions. Subgroups contain zero, one or multiple rules.



Screenshot 1: Application Menu

History

When working with the web application, certain actions of a user are captured. The history allows the user to quickly access these captured actions and so the history acts as a shortcut to allow quicker navigation.

Search

The search page allows to search for rules and/or actions. The name and description of the rules and actions is searched for the given term.

The results displayed allow to directly jump to the project and rulegroup where the rule or action is located.

Export File

Allows to export a project. The file created is a regular zip file and contains all rulegroups, subgroups, rules and actions for a given project. The resulting zip file can be used directly with the ruleengine to execute the rules.

Import File

Allows to import a project that previously has been exported. The user has to specify the name and description and the owner group of the imported project.

Checks

Rules use predefined checks to achieve their task. One rule uses exactly one check at a time. By combining rules with different checks a complex logic can be built.

An example for a check is „is equal to", „is greater than" or „is not null". Currently there are 40+ different checks available.

The list of checks provides links to a list of methods that are defined for each check. There is also a link to the documentation of the checks.

Actions

Actions allow to execute certain actions based on if the rule group as a whole failed or if it passed. Actions may update or calculate data or perform other tasks.

The list of actions provides links to a list of methods that are defined for each action and there is also a link to the documentation of the actions.

Files

Visible only for users in the *Admin* group. The web application uses Beanshell scripts and Apache Velocity templates to separate the programming logic of the web application from its representation on the screen. In the Files section a user with administrative rights can change the web application code or display on the fly.

Groups

Visible only for users in the *Admin* group. Groups define a collection of users that have the same access rights to projects. Different groups can contain different users and these can have access to multiple projects.

Users

Visible only for users in the *Admin* group. A user can be assigned to one or multiple groups. A user needs to authenticate to the application to be able to use it.

Configuration

Visible only for users in the *Admin* group. The configuration defines the details of the connection of the web application to the database and optional LDAP settings.

For the application to connect to the underlying database, it needs information on

- the database host – specify the hostname or IP adress
- the port used
- the database name
- the userid of a user with read and write privileges
- the users password

Specify an optional/additional folder where plugins belonging to the Pentaho PDI tool are located. When a Pentaho PDI transformation is attached to a project, for being able to determine the fields used in the transformation, the plugins used must be available to the web application. The web application will include this folder to scan for plugins.

You may use an LDAP server to verify the credentials of a user. Specify:

- LDAP server hostname or IP adress
- LDAP server port
- LDAP Server domain name

The process will first try to authenticate the user with the specified userid to the LDAP server and if this fails it will try to authenticate the user to the database. If no LDAP server is defined it will authenticate the user to the database only.

Documentation

A link to documents relevant to the web application and the ruleengine in general.

About

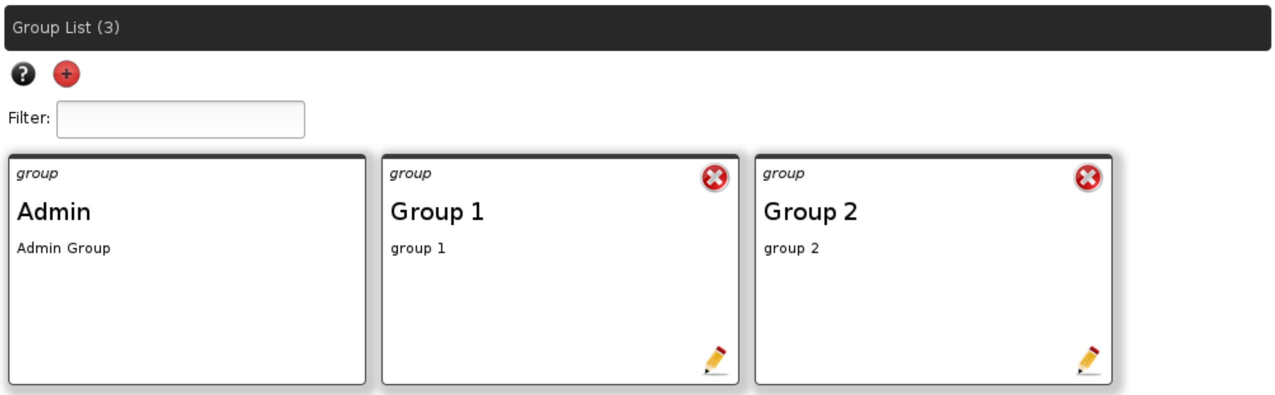
The about page shows information about the web application version and other details.

Concepts

This section explains the concepts that build the base for this web application. It is also related to the concept of the ruleengine that is used when the rules are executed against a set of data.

Group Management

The web application allows to define and modify groups. Users can be assigned to one or more groups. The settings for a project allow the definition to which group the project belongs. All users that are assigned to this group may read and update all details of the project, rulegroups, subgroups, rules and actions. Users not being part of the group have a read-only access to all the mentioned parts. The group *Admin* can not be updated or deleted, as it is required by the system. Users that are assigned to this group have admin privileges for the web application.


Group List (3)

Filter:

group
Admin
Admin Group

group
Group 1
group 1

group
Group 2
group 2

Screenshot 2: List of groups

User Management

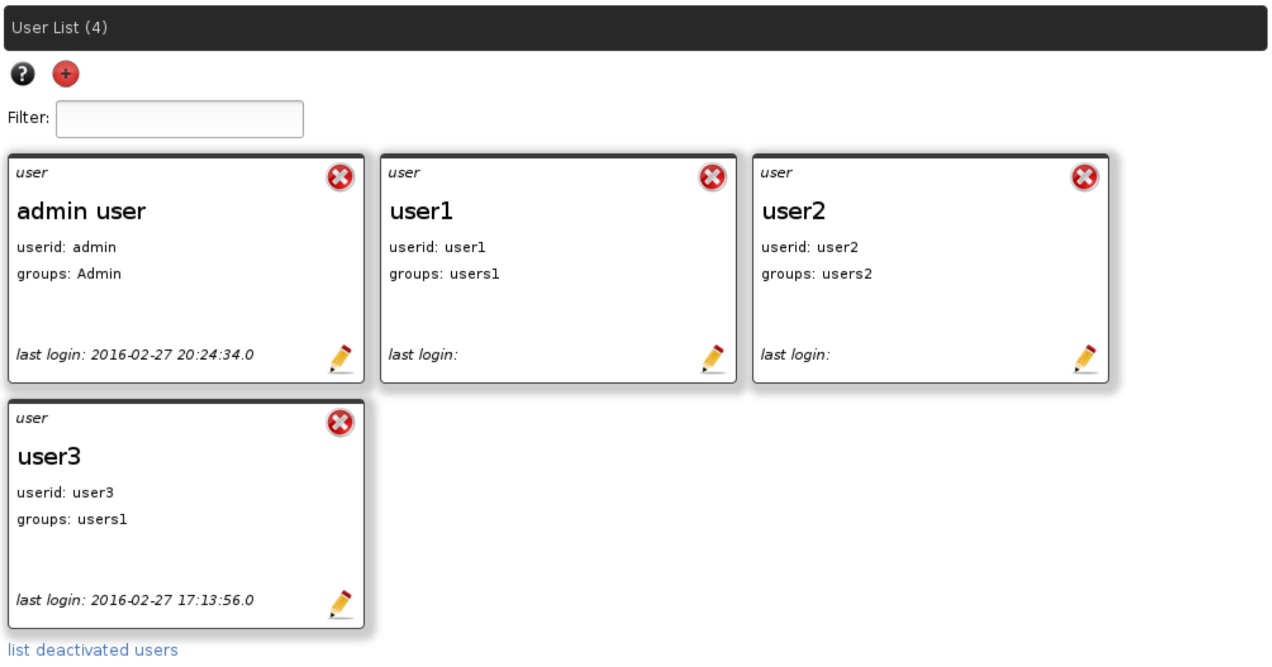
The web application allows to define and update users. Users can be assigned to one or multiple groups. See „*Group Management*" for more details.

Each user that is added will have access to the web application and the groups and in turn projects that are assigned to her/him.

Users may be deactivated. A deactivated user can not login to the web application anymore but all relevant data of the user such as history are conserved. When the user is finally deleted, all user related data is deleted.

Currently the users password is displayed once when the user is created (and never again). In a future release this will be changed to a more sophisticated approach.

In the configuration dialog a user with administrative privileges may configure the settings of a LDAP host. On successful setup, users will be authenticated against LDAP. Therefore the userid in this web application and the if of the user in LDAP have to be the same.


User List (4)

Filter:

user	user	user
admin user	user1	user2
userid: admin	userid: user1	userid: user2
groups: Admin	groups: users1	groups: users2
last login: 2016-02-27 20:24:34.0	last login:	last login:

user
user3
userid: user3
groups: users1
last login: 2016-02-27 17:13:56.0

list deactivated users

Screenshot 3: List of users

Project Management

Projects are containers for rulegroups, subgroups, rules and actions. They group all parts together. When a project is created it has to be assigned to a group. Users assigned to this group have read and write access to all parts of the project. Other users have read-only access to all parts of the project.

If a project is marked as private project, only users assigned to the group the project belongs to have access to the project. Other users don't see the project.

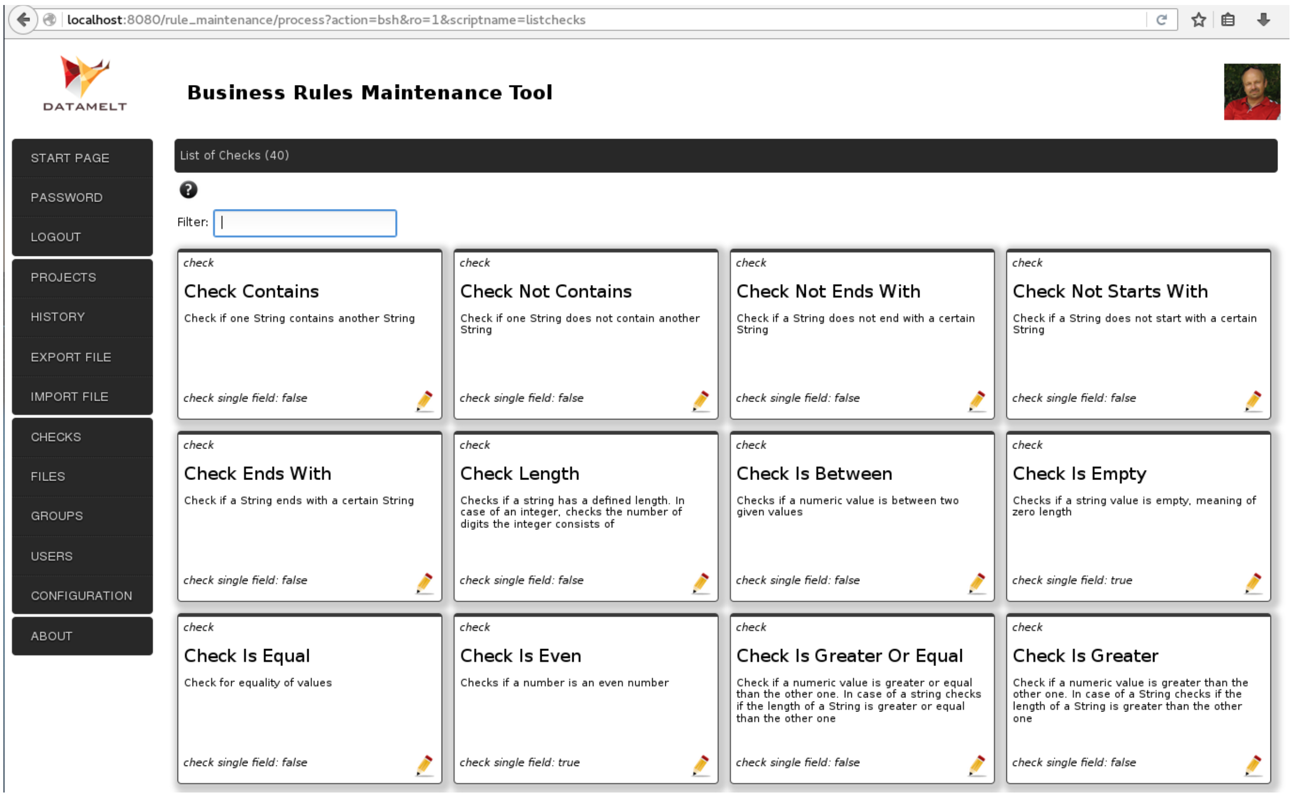
Projects may be copied. All content of the project is copied into a new project. Also, projects may be exported into a zip file and they also may be imported back into a new project.

Checks

A check is an individual module that is designed to fulfill one specific test. For example one

check “is equal to”. It tests, if the given data is equal to the expected result.
One single check usually has the ability to use different types of data such as textual data, data of type date or numeric data.

There are currently 40 checks available. They are referenced by the rules and by the combination of several checks a complex logic can be built to test for certain conditions of the data.



check
Check Contains
Check if one String contains another String
check single field: false

check
Check Not Contains
Check if one String does not contain another String
check single field: false

check
Check Not Ends With
Check if a String does not end with a certain String
check single field: false

check
Check Not Starts With
Check if a String does not start with a certain String
check single field: false

check
Check Ends With
Check if a String ends with a certain String
check single field: false

check
Check Length
Checks if a string has a defined length. In case of an integer, checks the number of digits the integer consists of
check single field: false

check
Check Is Between
Checks if a numeric value is between two given values
check single field: false

check
Check Is Empty
Checks if a string value is empty, meaning of zero length
check single field: true

check
Check Is Equal
Check for equality of values
check single field: false

check
Check Is Even
Checks if a number is an even number
check single field: true

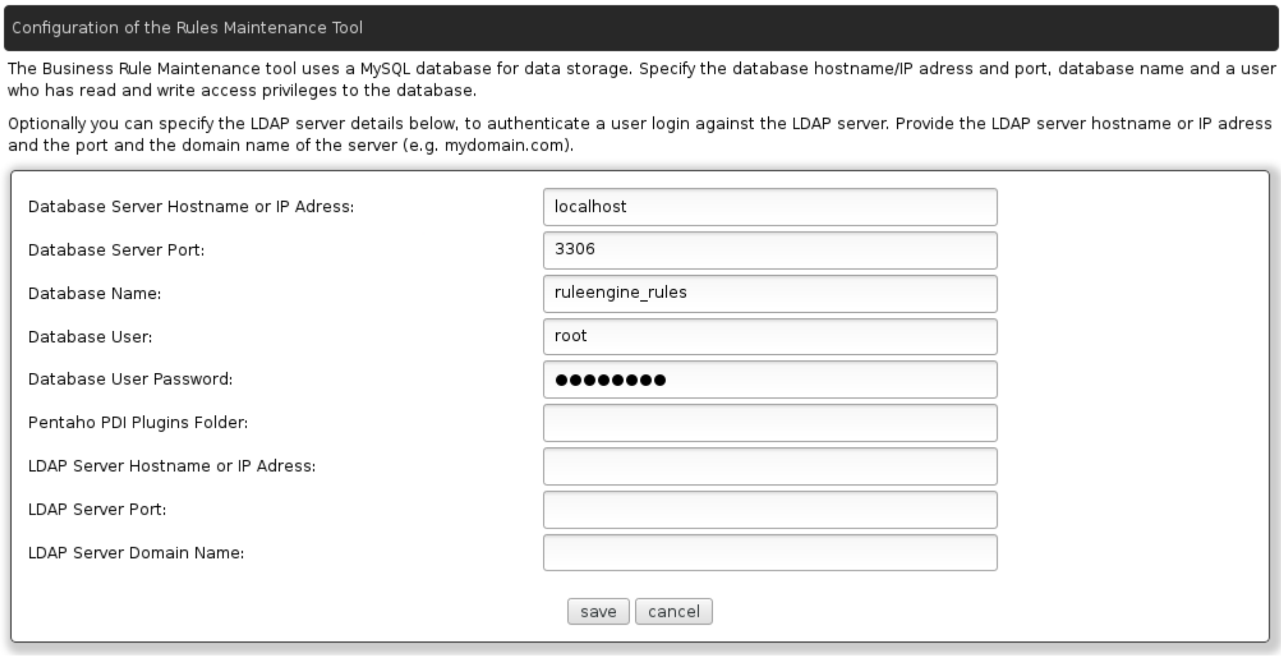
check
Check Is Greater Or Equal
Check if a numeric value is greater or equal than the other one. In case of a string checks if the length of a String is greater or equal than the other one
check single field: false

check
Check Is Greater
Check if a numeric value is greater than the other one. In case of a String checks if the length of a String is greater than the other one
check single field: false

Screenshot 4: List of checks

The ruleengine can be extended by other checks. Please read the appropriate documentation on how to create your own checks.

Configuration



Configuration of the Rules Maintenance Tool

The Business Rule Maintenance tool uses a MySQL database for data storage. Specify the database hostname/IP adress and port, database name and a user who has read and write access privileges to the database.

Optionally you can specify the LDAP server details below, to authenticate a user login against the LDAP server. Provide the LDAP server hostname or IP adress and the port and the domain name of the server (e.g. mydomain.com).

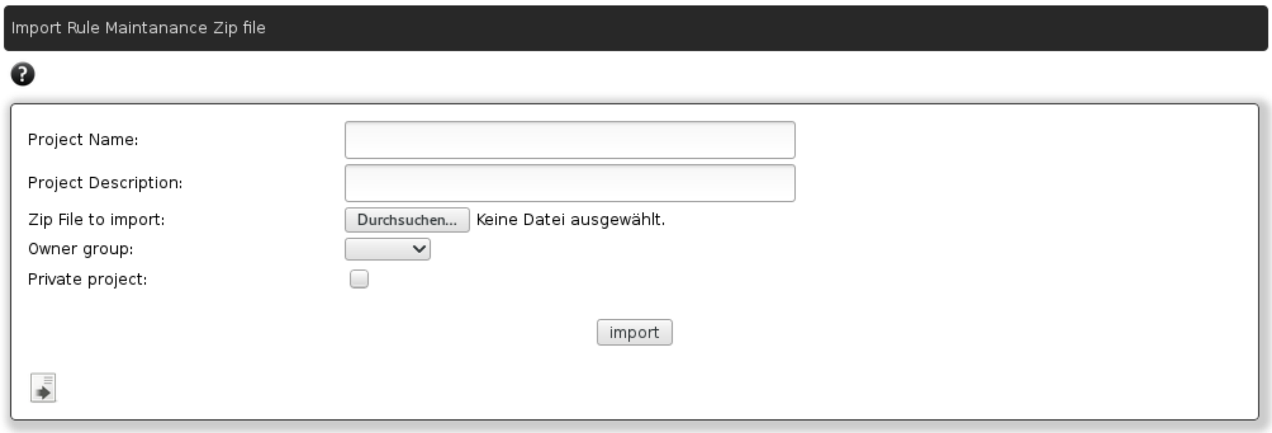
Database Server Hostname or IP Adress:	localhost
Database Server Port:	3306
Database Name:	ruleengine_rules
Database User:	root
Database User Password:	●●●●●●●●●
Pentaho PDI Plugins Folder:	
LDAP Server Hostname or IP Adress:	
LDAP Server Port:	
LDAP Server Domain Name:	

Screenshot 5: Configuration

The configuration page allows to define the basic settings for the application. An administrator may update the settings for accessing the database and the LDAP settings.

Import Project

A project – in the form of a zip file - that has previously been exported using this web application can be imported. This allows the exchange of project files and definitions between different users or across different companies.



Import Rule Maintanance Zip file

Project Name:	
Project Description:	
Zip File to import:	Durchsuchen... Keine Datei ausgewählt.
Owner group:	
Private project:	☐

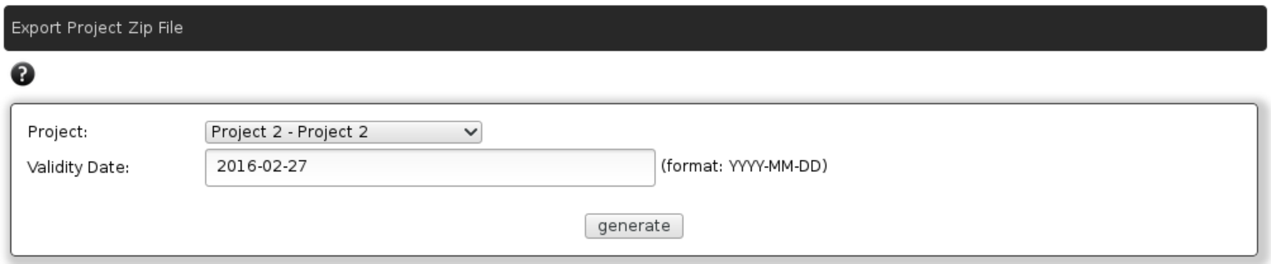
Screenshot 6: Import project

If a project is exported to a file, it can be safely deleted and then re-created by importing the project file. There is no data loss in this process.

Export Project

A project may be exported. In the process a zip file is created that can directly (without modifications) be run with the JaRE ruleengine – either standalone, using Pentaho PDI or running the ruleengine serverside.

When exporting, the user has to select the project and enter a validity date for which he/she wants to do the export. The validity date steers which rulegroups are exported. Rulegroups have a start and end validity. Only those rulegroups are exported which a valid for the specified validity export date.



Project: Project 2 - Project 2
Validity Date: 2016-02-27 (format: YYYY-MM-DD)

Screenshot 7: Export project

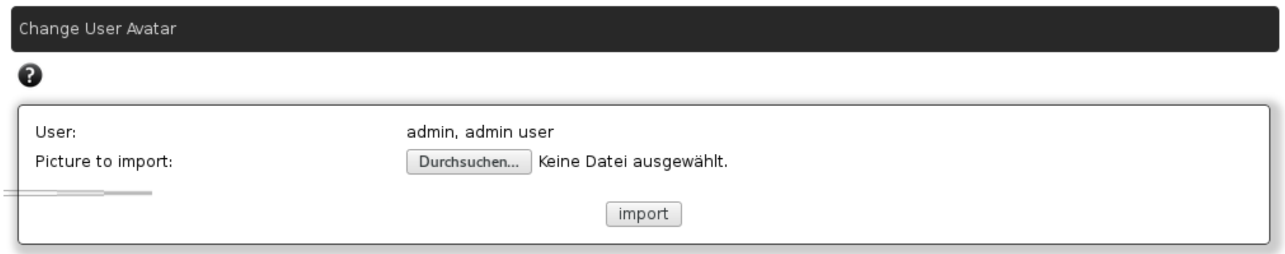
History

Projects can contain a certain number of rulegroups, subgroups, rules and actions. This makes it time consuming to locate a certain item. The History lists components that lately have been updated by the user that is logged in. The user won't see the history of other users.

An entry in the history can be clicked to directly jump to the project or rulegroup where the entry is located.

User Avatar

The user may choose a picture for his/her representation. It is shown on the upper right-hand side of the web application. Clicking on the picture – or the default picture if undefined – will allow the user to upload a picture to be used as the users avatar.



User: admin, admin user

Picture to import: Keine Datei ausgewählt.

Screenshot 8: Change User avatar

Help

In various places/screens of the web application, there is help available identified by a little icon with a question mark inside:

?

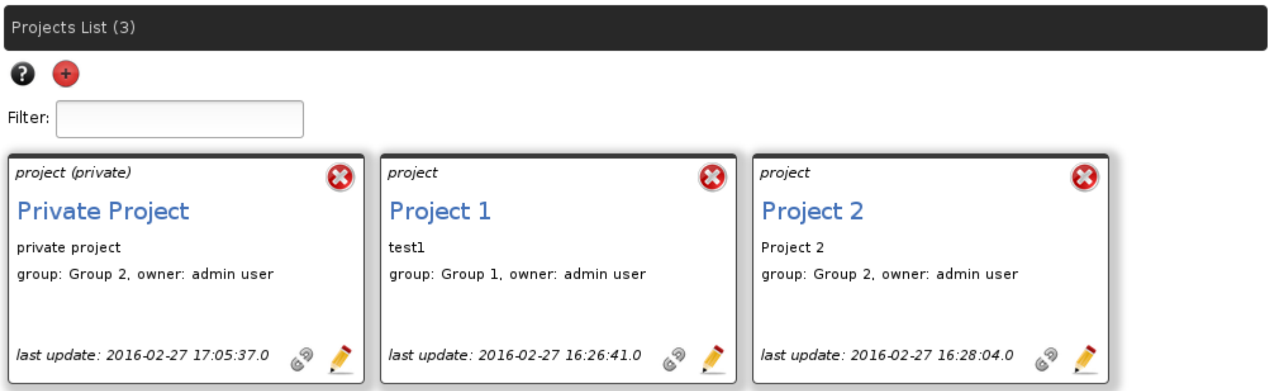
Click on the icon to display a short help text. Click on it again to hide the help text.

Projects

Project is the highest item in the hierarchy. A project may contain zero, one or multiple rulegroups. Rulegroups may contain zero, one or multiple subgroups. Rulegroups also may contain zero, one or multiple actions. Subgroups contain zero, one or multiple rules.

Project belong to exactly one owner group. All members of this group can read, add, update and delete all content of the project. If the project is marked as being “private“, users that are not members of the owner group can not see the project.

If you are a user of Pentaho PDI then you can attach a ETL (Extract – Transform – Load) transformation to a project. The advantage is, that instead of typing in field names and types, you will get a list of fields for selection, which the web application extracts directly from the ETL transformation.


Projects List (3)

Filter:

project (private)	project	project
Private Project	Project 1	Project 2
private project	test1	Project 2
group: Group 2, owner: admin user	group: Group 1, owner: admin user	group: Group 2, owner: admin user
last update: 2016-02-27 17:05:37.0	last update: 2016-02-27 16:26:41.0	last update: 2016-02-27 16:28:04.0

Screenshot 9: List of projects

Rulegroups

Rulegroups are the containers for subgroups, rules and actions. They have a validity start and end date defined. It defines in which timeframe a certain rulegroup and all its content is valid. This allows to define a logic that is automatically valid at a certain date - so you can plan ahead - and later on automatically expires and is not used anymore.

Subgroups

Subgroups are the containers for rules. Within one subgroup all rules are connected using the same condition: either "and" or "or" - you can not use a mixture. Subgroups are connected to each other (to the previous subgroup) also either by "and" or "or". The combination of "and" or "or" in the subgroup plus the combination of "and" or "or" between the subgroups gives the user the flexibility to define any complex logic.

Rules

The rules are the working horses of the ruleengine. On execution data is passed to the rule, the rule is executed and the result is a "passed" or "failed" state. All rules in a rulegroup form a certain logic using "and" or "or" conditions.

Actions

Actions belong to the rulegroup they are defined in. When a rulegroup – and all its rules – is executed, the result is either a "passed" or "failed" state. An action can run based on this state and execute e.g. an update to the data, do calculations and modify the data, send messages, log information and more.

A rulegroup may have zero, one or many actions. The actions are executed one after the other.

Pentaho PDI (Kettle)

The web application produces/exports a project and all relevant settings and data to a single zip file. This file can be used with the JaRE ruleengine. Note that it is **not** required to use any Pentaho software to do so – the ruleengine can run standalone from the command line or integrated into any Java application.

But there is a certain integration of the ruleengine with the ETL tool from Pentaho which is called *Pentaho PDI* or also *Pentaho Kettle* (previous name). There are plugins available for *Pentaho PDI* which make it very easy to use and interact with the ruleengine. This helps the ETL developer and designer to separate the code from business rules and allows for cleaner design and code.

The ruleengine plugins for Pentaho PDI are available through the Pentaho marketplace. They can easily be installed through the interface of the ETL tool.

The plugins in PDI are what I call "puzzle pieces". A certain number of these pieces form a proper picture and the ruleengine plugin can be one of it. The plugin simplly references the project zip file created with the web application and in turn executes the rules using the JaRe – Java Rule Engine.

The output will be twofold: An incoming row will be checked and it will be passed to the next plugin or step as one row. It will contain additional fields from the ruleengine indicating how many rule groups and rules failed. Optionally there is an additional second output stream, that shows the details of each row, compared to each rule that ran.

Within PDI the user can then decide how to continue based on the results of the ruleengine.

Workflow

Below find some details and guidelines for a standard workflow that can be used.

Preparations

Any user that is able to login to the application can create projects.

It is advisable that you think about following a naming convention for projects, rulegroups, subgroups, rules and actions. This will ensure that you or others are able to understand the purpose more easily. Also, when running the project against the ruleengine, some of the results will reference the names or descriptions you have assigned to the individual elements.

Create a project

In general the name should be a rather short text string.- a sort of identifier for the element. The description may be longer.

Assign the project to a group. Users who are members in this group will be able to view and update the project content.

If you do not want that users outside this group see this project, then mark the project as private.

Save your settings and click on the close button to go back to the previous page.

Create a rulegroup

Back on the project list page, click on the name of the project. On the following page add a new rulegroup by clicking the relevant icon in the top part.

The rulegroup will contain subgroups, rules and actions. These form a logical unit for a specific condition or multiple ones you want to test.

Again, select a rather short name – according to your naming convention – and enter a more detailed description. Enter a valid from and a valid until date for the rulegroup. You ma select a future in the past for the valid from date. If you think your group should be valid forever, simply added or select a date far in the future for the valid until date. Such as 2099-12-31.

Save your settings and click on the close button to go back to the previous page.

When you create a rulegroup one subgroup inside the rulegroup is created automatically. It is named „subgroup_1“. You can edit this subgroups details if necessary. Take special attention to the rules connector displayed in the lower left hand corner of the subgroup. It defines how the rules in this subgroup are connected to each other. The values can be „and“ or „or“.

Later in the process you can create more subgroups as required.

Create a rule

Inside the subgroup displayed click to add a new rule.

Enter a name and description for the rule. Next, enter the name of the field that shall be checked (tested) and next to it the data type of the data. This references a field of the data source you will later run the rules against. It could be a CSV file or a database or some other data source. The field name should be entered exactly as it is in the data source; uppercase/lowercase may be relevant.

Next, select a check you want to use against the field defined before. E.g. a check for equality. This is a good moment to look at the documentation for the checks to find out if the desired check is available and if the logic you want to use is implemented. Mainly you want to make sure that the data types involved are usable for a certain check.

Next there are two possibilities: Either check the field against another field of the data source. Or check the field against a given fixed value. If you check against another field, enter the field name and select it's type. If not, leave this definition empty and enter in the next line the value you want to check against and it's type.

Some of the checks allow you to specify additional parameters. For example the check for equality of strings takes an optional parameter, if the comparison should be case senitive. If you need to specify an additional parameter, enter it and it's type.

Next come the message that is generated in case the data passes the check and the message in case it fails. You may enter a free text to describe the two facts. Next to the text field to enter the message, there is a button. It auto-generates a message for you.

In the auto-generated message there will be placeholders: $1 and $0 (sometimes only one of them). The placeholder $1 represents the actual value of the data. The placeholder $0 represents the value to compare against – either the data from another field or the given fixed value.

At this point the rule is complete. Save it and click „close“ to go back to the rulegroup listing.

More rules

If you want to implement more rules, repeat the last step. The display rules connector of the subgroup will indicate if the rules in a certain subgroup are connected by an „and“ or „or“ condition. You can change this by editing the subgroup.

More subgroups

Subgroups may also be connected by an „and“ or „or“ condition. This condition always defines the connection to the **previous** subgroup. If you have two subgroups and „Connector to previous group“ (edit the subgroup to change) is set to „or“, then the logic is such that the subgroup 1 has to pass the checks or the subgroup 2 has to pass the checks. The other way around, if they are connected with an „and“ condition, then both subgroups have to pass the checks.

So all rules in the individual subgroups and the subgroups itself together form a logic. This collection of logic is captured inside a rulegroup. The ruleengine will execute the rules in the rulegroup and the rulegroup will pass or fail the tests.

Create an action

Actions belong to the rulegroup they are defined in. Based on the fact if the rulegroup passes or fails, an action – or multiple ones - may be executed.

Enter a name and description for the action. Next, select if the action should be executed when the rulegroup fails or when it passes. Next, select a field and it's type that shall be updated. The next step is to select which action to execute – select one from the dropdown. Make sure you select the correct action for the data types involved.

Again, a good moment to consult the documentation for the actions to determine if the desired action is available, for which data types it is applicable and what (optional) additional parameters are available.

As with the rules, there are two possibilities. An action can use the data of another field or a given fixed value that it will use to execute the action. If the data comes from a field, enter the field name and select it's type.

If you are using a fixed value, then leave the field to retrieve data from and it's type empty and enter the fixed value in the first textbox for the parameters. And select the type of the parameter's data.

Specify any additional parameters in the following textboxes and select the data type for each parameter.

Finally save the action and click on „close" to go back to the rulegroup listing.

More actions

Continue to create more actions as required. Actions will be executed one after the other as defined in the Business Rule Maintenance web application.

Export project

Once you have defined all rules and actions that are required to check a certain business logic, export the project. This will create a zip file that contains all your definitions in a single file. This file can directly be run with the ruleengine.

In the menu click the „Export Project" entry. Select the relevant project from the dropdown list. Next, select a validity date. As described further above, rulegroups (which contain all rules and actions) have a defined validity in terms of time. These are valid from a given date and until a given date. When you select the validity date for the project to be exported, only those rulegroups will be exported, which are valid at the given validity date. The other way around: rulegroups, that are not valid at the given date (expired), will not be included in the project zip file.

This feature allows you to plan changes in your business logic ahead of time. You can implement new or changed logic well before it will becomes active.

Executing rules and actions

The final step of the workflow is then to execute the defined rules and actions.

One way is to run them in the Pentaho ETL tool where you will get the results of the execution directly inside the ETL tool.

You may also include the ruleengine in your own software or web application and run it from there. The API for the ruleengine will list the possible methods to retrieve the results of the execution and the messages that were generated.

Another possibility is to run the ruleengine on a separate server. It then waits for a socket connection from a client to execute the rules and actions.

Software Updates

The Business Rules Maintenance web application uses several software libraries. If a newer version of these libraries is available it might be possible to replace the existing ones. This depends on the compatibility of the newer libraries with this web application code.

The libraries are located in following folder:

[Tomcat root folder]/webapps/rule_maintenance/WEB-INF/lib

Updates to the base components of Java, Apache Tomcat and MySQL/MariaDB should in general be possible without interference to the web application. Otherwise check if a newer version of the web application is available.

Extending the Ruleengine

The ruleengine can be extended for the two main concepts: checks and actions. By implementing a defined interface the developer can create new checks or actions. The resulting Java classes can then be added to the ruleengine jar file or to the Java classpath, so they can be located.

It is of course also possible to added new extensions to the GitHub repository, so they will be reviewed and then included in a future release of the application. This way the community will also benefit from it.

Contact

For questions or feedback please contact me.

- Email: uwe.geercken@web.de
- Twitter: @uweeegeee

Links

Below is a list of links to software or web pages references in this document.

- MySQL/MariaDB database schema:

  Link: https://github.com/uwegeercken/rule_maintenance_db

  Download file: ruleengine_rules.sql

- Business Rules Maintenance web application .war file

  Link: https://github.com/uwegeercken/rule_maintenance_war

  Download file: rule_maintenance.war

- Apache Velocity Template engine

  Link: http://velocity.apache.org/

- MySQL Database Server and tools

  Link: http://www.mysql.com/

- MariaDB Database Server and tools

  Link: http://mariadb.org/

- Oracle Java

  Link: https://www.java.com

Defaults

Listed below are defaults that the web application uses.

- Business Rules Maintenance application default login

  userid: *admin*

  password: *admin*

- Database name: *ruleengine_rules*

Subject Index