Business Rule Engine

Detailed Listing of Actions

for Jare Version 0.84

Table of Contents

Overview

This document contains a detailed listing of the available actions for the Business Rule Engine „Jare“. It lists the available data types, their combinations and possible optional or required parameters.

Actions are defined on the rulegroup level. When a rulegroup fails or passes then none, one or multiple actions can be executed. Actions are executed in the sequence that they have been defined. It is also possible to define actions that are execute when the rulegroup passes and actions when the rulegroup fails at the same time (the actions is executed in any case).

A condition must be specified when to execute the action – when the rulegroup fails or passes or both. Additionally a field - and it's type - must be specified that will be updated by the action.

Sometimes an action will require additional values to be passed to it. This can either be the value of another field or a defined fixed value.

The values selected in the action from “Field to retrieve data”, “Parameter 1”, “Parameter 2” and “Parameter 3” are passed to the action in this exact sequence. The “Field to retrieve data” is optional and indicates from which other field a value should be retrieved. It can be specified but must not be.

If for example the action requires two values (arguments/parameters) then either “Field to retrieve data” and “Parameter 1” have to be specified or alternatively “Parameter 1” and “Parameter 2” (in this case “Field to retrieve data” must be left blank). The “Field to update” will be updated with the resulting value.

The definition for “Interface Name” in the listed tables further below, indicates the name of the action in the Business Rules Maintenance Tool. This is a web application avaiable to orchestrate brusiness logic – including actions – using an user-oriented interface.

Prerequisites

The default date format used is: yyyy-MM-dd (four digits for the year, two digits for the month and two digits for the day of the month). If not otherwise specified for String to Date conversions, it is assumed that the value is provided in this format. Some actions allow to specify a different format for the date than the default one. In this case the date formats need to be according to the Java SimpleDateFormat Class format definition.

All data types listed here are Java related data types.

Add Leading Spaces

Purpose:	Add leading spaces to a value until a given length of the value is reached
Java Class:	StringAction
Interface Name:	add leading spaces

Applicable to:

Value 1	**Value 2**	**Return Type**
String	Integer	String
Note:		
Optional Value(s)	**Explanation**	

Add Leading Zeros

Purpose:	Add leading zeros to a value until a given length of the value is reached
Java Class:	StringAction
Interface Name:	add leading zeros

Applicable to:

Value 1	**Value 2**	**Return Type**
String	Integer	String
Note:		
Optional Value(s)	**Explanation**	

Append Value

Purpose:	Append a given value to the end of another value
Java Class:	StringAction
Interface Name:	append

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
String	Integer	String
String	Long	String
Note:		
Optional Value(s)	**Explanation**	
Value 3: String	Separator to be used between the values	

Prepend Value

Purpose:	Prepend a given value to the start of another value
Java Class:	StringAction
Interface Name:	prepend

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
String	Integer	String
String	Long	String
Note:		
Optional Value(s)	**Explanation**	
Value 3: String	Separator to be used between the values	

Trim Value

Purpose:	Remove all leading and trailing space characters from a String
Java Class:	StringAction
Interface Name:	trim

Applicable to:

Value 1	**Value 2**	**Return Type**
String		String
Note:		
Optional Value(s)	**Explanation**	

Concatenate Values

Purpose:	Concatenate two values
Java Class:	StringAction
Interface Name:	append value

Applicable to:

Value 1	**Value 2**	**Return Type**
String	Double	String
String	Float	String
String	Long	String
String	Integer	String
String	String	String
Note:		
Optional Value(s)	**Explanation**	
Value 3: String	Separator to be used between the value	

Lowercase Value

Purpose:	Change a value to it's lowercase representation
Java Class:	StringAction
Interface Name:	lowercase

Applicable to:

Value 1	**Value 2**	**Return Type**
String		String
Note:		
Optional Value(s)	**Explanation**	

Uppercase Value

Purpose:	Change a value to it's uppercase representation
Java Class:	StringAction
Interface Name:	uppercase

Applicable to:

Value 1	**Value 2**	**Return Type**
String		String
Note:		
Optional Value(s)	**Explanation**	

Replace Value

Purpose:	Replace a given value – or part of it - with another value
Java Class:	StringAction
Interface Name:	replace value

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 has to contain a regular expression. All found occurrences of the regular expression will be replaced with value 3.		
Required Value(s)	**Explanation**	
Value 3: String	the replacement for the original value or parts of it	

Replace Value from Mapping File

Purpose:	Replace a given value with a value from a mapping file. The given value will be looked up in the mapping file and replaced with the value found in the mapping file.
Java Class:	StringAction
Interface Name:	replace value from map

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 has to contain the full path and filename of the mapping file.		
Required Value(s)	**Explanation**	

Replace Value from List

Purpose:	Replace a given value with a value from a list of values separated by a comma. The integer values is used to retrieve the corresponding element form the list by it's index.
Java Class:	StringAction
Interface Name:	replace value from list

Applicable to:

Value 1	**Value 2**	**Return Type**
Integer	String	String
Note: Value 2 is a list of values separated by comma.		
Required Value(s)	**Explanation**	

Mask Value

Purpose:	Masks a given value with a defined character. The optional integer values are used to define the start and/or end position of the given value that shall be masked.
Java Class:	StringAction
Interface Name:	mask value

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 is a list of values separated by comma.		
Optional Value(s)	**Explanation**	
Value 3: Integer	The start position for masking	
Value 4: Integer	The end position for masking	

Encrypt Value

Purpose:	Encrypts a given value with a defined key. The returned value will be a hexadecimal string.
Java Class:	StringAction
Interface Name:	encrypt value

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 is the encryption key		
Optional Value(s)	**Explanation**	

Decrypt Value

Purpose:	Decrypts a given value with a defined key. Value 1 must be a hexadecimal string that was produced by the encrypt action and using the same key (Value 2) as was used for the encryption.
Java Class:	StringAction
Interface Name:	decrypt value

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 is the decryption key		
Optional Value(s)	**Explanation**	

Set Value (String)

Purpose:	Sets a given value of a string
Java Class:	StringAction
Interface Name:	set value (string)

Applicable to:

Value 1	**Value 2**	**Return Type**
String		String
Note:		
Optional Value(s)	**Explanation**	

Set Value (Date)

Purpose:	Sets a given value of a date
Java Class:	DateAction
Interface Name:	set value (date)

Applicable to:

Value 1	**Value 2**	**Return Type**
Date		Date
String		Date
Note:		
Optional Value(s)	**Explanation**	
Value 2: String	In case value 1 is a String, this value optionally defines the date format of the date.	

Set Value (Date)

Purpose:	Sets a given value of a date and returns it as a String
Java Class:	DateAction
Interface Name:	set value (date)

Applicable to:

Value 1	**Value 2**	**Return Type**
Date	String	String
Note: The String value is the format for the returned value according to the Java SimpleDateFormat class.		
Optional Value(s)	**Explanation**	

Set Value (Date)

Purpose:	Sets a given value of a date and returns it as a String
Java Class:	DateAction
Interface Name:	set value (date)

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 1 is a date represented as a String value. Value 2 describes the format of Value 1.		
Required Value(s)	**Explanation**	
Value 3	The format for the returned value according to the Java SimpleDateFormat class.	

Set Value (Number)

Purpose:	Sets a given value of a number
Java Class:	MathAction
Interface Name:	absolute

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Float		Float
Long		Long
Integer		Integer
Boolean		Boolean
BigDecimal		BigDecimal
Note:		
Optional Value(s)	**Explanation**	

Substring Value

Purpose:	Sets the value to a given substring
Java Class:	StringAction
Interface Name:	set value (String)

Applicable to:

Value 1	**Value 2**	**Return Type**
String	Integer	String
Note: Value 2 defines from which position of the value the substring will start		
Optional Value(s)	**Explanation**	
Value 3: Integer	Defines at which position the substring will end	

Applicable to:

Value 1	**Value 2**	**Return Type**
String	String	String
Note: Value 2 defines the String value until which the substring shall extend		
Optional Value(s)	**Explanation**	

Set Today's Date

Purpose:	Set the value to the current date
Java Class:	DateAction
Interface Name:	set today date

Applicable to:

Value 1	**Value 2**	**Return Type**
String		String
Note: Value 1 contains the date format to use		
Optional Value(s)	**Explanation**	
Value 2: Integer	Define an offset of days – positive or negative – that will be added to the current date	

Set Last Day of Month

Purpose:	Set the value to the last day of the month
Java Class:	DateAction
Interface Name:	set last day of month

Applicable to:

Value 1	**Value 2**	**Return Type**
		Date
Integer	Integer	String
Note: You can specify the year and month as integer values for which to calculate the last day.		
Optional Value(s)	**Explanation**	
Value 1: Date	Specify a date for which the last day of the month is calculated	

Set First Day of Month

Purpose:	Set the value to the first day of the month
Java Class:	DateAction
Interface Name:	set first day of month

Applicable to:

Value 1	**Value 2**	**Return Type**
		Date
Integer	Integer	String
Note: You can specify the year and month as integer values for which to calculate the first day.		

Optional Value(s)	**Explanation**
Value 1: Date	Specify a date for which the first day of the month is calculated

Set Mid Day of Month

Purpose:	Set the value to the mid day of the month – day 15
Java Class:	DateAction
Interface Name:	set mid day of month

Applicable to:

Value 1	**Value 2**	**Return Type**
		Date
Integer	Integer	String
Note: You can specify the year and month as integer values for which to calculate the mid day.		
Optional Value(s)	**Explanation**	
Value 1: Date	Specify a date for which the mid day of the month is calculated	

Absolute Value

Purpose:	Calculates the absolute value of a given value
Java Class:	MathAction
Interface Name:	absolute

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Float		Float
Long		Long
Integer		Integer
Note:		
Optional Value(s)	**Explanation**	

Acosinus Value

Purpose:	Calculates the acosinus of a given value
Java Class:	MathAction
Interface Name:	acosinus

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Cosinus Value

Purpose:	Calculates the cosinus of a given value
Java Class:	MathAction
Interface Name:	cosinus

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Cosinush Value

Purpose:	Calculates the cosinush of a given value
Java Class:	MathAction
Interface Name:	cosinush

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Asinus Value

Purpose:	Calculates the asinus of a given value
Java Class:	MathAction
Interface Name:	asinus

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Sinus Value

Purpose:	Calculates the sinus of a given value
Java Class:	MathAction
Interface Name:	sinus

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Sinush Value

Purpose:	Calculates the sinush of a given value
Java Class:	MathAction
Interface Name:	sinush

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Atangens Value

Purpose:	Calculates the atangens of a given value
Java Class:	MathAction
Interface Name:	atangens

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Tangens Value

Purpose:	Calculates the tangens of a given value
Java Class:	MathAction
Interface Name:	tangens

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Tangensh Value

Purpose:	Calculates the tangensh of a given value
Java Class:	MathAction
Interface Name:	tangensh

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Note:		
Optional Value(s)	**Explanation**	

Square Root Value

Purpose:	Calculates the square root of a given value
Java Class:	MathAction
Interface Name:	square root

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Float		Double
Integer		Double
Long		Double
Note:		
Optional Value(s)	**Explanation**	

Square Value

Purpose:	Calculates the square of a given value
Java Class:	MathAction
Interface Name:	square

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Double
Float		Double
Long		Long
Integer		Long
Note:		
Optional Value(s)	**Explanation**	

Add Values

Purpose:	Calculates the sum of two values
Java Class:	MathAction
Interface Name:	sum

Applicable to:

Value 1	**Value 2**	**Return Type**
Double	Double	Double
Float	Float	Float
Integer	Integer	Integer
Integer	Long	Long
Long	Integer	Long
Long	Long	Long
Note:		
Optional Value(s)	**Explanation**	

Subtract Values

Purpose:	Subtracts two values
Java Class:	MathAction
Interface Name:	subtract

Applicable to:

Value 1	**Value 2**	**Return Type**
Date	Date	Long
Double	Double	Double
Float	Float	Float
Integer	Integer	Integer
Integer	Long	Integer
Long	Integer	Long
Long	Long	Long
Note: In case of subtracting dates, the result will be given in seconds.		
Optional Value(s)	**Explanation**	

Add Percentage

Purpose:	Add a percentage of a value to the value
Java Class:	MathAction
Interface Name:	add percentage

Applicable to:

Value 1	**Value 2**	**Return Type**
Double	Double	Double
Double	Float	Double
Double	Integer	Double
Integer	Double	Double
Integer	Float	Double
Integer	Integer	Double
Long	Double	Double
Long	Float	Double
Long	Integer	Double
Note:		
Optional Value(s)	**Explanation**	

Subtract Percentage

Purpose:	Subtract a percentage of a value from the value
Java Class:	MathAction
Interface Name:	subtract percentage

Applicable to:

Value 1	**Value 2**	**Return Type**
Double	Double	Double
Double	Float	Double
Double	Integer	Double
Integer	Double	Double
Integer	Float	Double
Integer	Integer	Double
Long	Double	Double
Long	Float	Double
Long	Integer	Double
Note:		
Optional Value(s)	**Explanation**	

Devide Values

Purpose:	Calculates the devision of two values
Java Class:	MathAction
Interface Name:	devide

Applicable to:

Value 1	**Value 2**	**Return Type**
BigDecimal	BigDecimal	BigDecimal
Double	Double	Double
Double	Integer	Double
Float	Float	Float
Integer	Integer	Double
Integer	Long	Double
Long	Integer	Double
Long	Double	Double
Long	Long	Double
Note:		
Optional Value(s)	**Explanation**	

Multiply Values

Purpose:	Multiplies two values
Java Class:	MathAction
Interface Name:	multiply

Applicable to:

Value 1	**Value 2**	**Return Type**
Double	Double	Double
Double	Integer	Double
Float	Float	Float
Integer	Integer	Long
Integer	Long	Long
Long	Double	Double
Long	Integer	Long
Long	Long	Long
Note:		
Optional Value(s)	**Explanation**	

Round Value

Purpose:	Rounds a given value
Java Class:	MathAction
Interface Name:	round

Applicable to:

Value 1	**Value 2**	**Return Type**
Double		Long
Float		Integer
Note:		
Optional Value(s)	**Explanation**	

Random Value

Purpose:	Generates a random integer value
Java Class:	MathAction
Interface Name:	round value

Applicable to:

Value 1	**Value 2**	**Return Type**
Integer	Integer	Integer
Float		Integer
Note: specify a minimum and maximum for the integer to be generated		
Optional Value(s)	**Explanation**	

Remainder Value

Purpose:	Returns the remainder value (also called modulo) of two values
Java Class:	MathAction
Interface Name:	remainder value

Applicable to:

Value 1	**Value 2**	**Return Type**
Integer	Integer	Integer
Long	Integer	Long
Long	Long	Long
Note:		
Optional Value(s)	**Explanation**	

Add Minutes

Purpose:	Adds a given number of minutes to a date
Java Class:	DateAction
Interface Name:	add minutes

Applicable to:

Value 1	**Value 2**	**Return Type**
Date	Long	Date
Note:		
Optional Value(s)	**Explanation**	

Convert to Integer

Purpose:	Converts a given String to an integer value
Java Class:	ConvertAction
Interface Name:	convert to integer

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Integer
Note: If the string value can not be converted, the value 0 is returned.		
Optional Value(s)	**Explanation**	

Convert to Long

Purpose:	Converts a given String to a long value
Java Class:	ConvertAction
Interface Name:	convert to long

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Long
Note: If the string value can not be converted, the value 0 is returned.		
Optional Value(s)	**Explanation**	

Convert to Double

Purpose:	Converts a given String to a double value
Java Class:	ConvertAction
Interface Name:	convert to double

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Double
Note: If the string value can not be converted, the value 0.0 is returned.		
Optional Value(s)	**Explanation**	

Convert to Float

Purpose:	Converts a given String to a float value
Java Class:	ConvertAction
Interface Name:	convert to float

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Float
Note: If the string value can not be converted, the value 0.0 is returned.		
Optional Value(s)	**Explanation**	

Convert 4 Characters Time

Purpose:	Converts a given String of four characters which represent time in the format *hhmm* to an integer value, representing total minutes. Example: value= 0133 = 1 hour, 33 Minutes = 93 Minutes
Java Class:	ConvertAction
Interface Name:	Convert 4 characters time

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Integer
Note: If the string value can not be converted, the value 0 is returned.		
Optional Value(s)	**Explanation**	

Convert 5 Characters Time

Purpose:	Converts a given String of four characters which represent time in the format *hh:mm* to an integer value, representing total minutes. The third character is the delimiter (any) between hours and minutes. Example: value= 30:26 = 30 hours, 26 Minutes = 1826 Minutes
Java Class:	ConvertAction
Interface Name:	Convert 5 characters time

Applicable to:

Value 1	**Value 2**	**Return Type**
String		Integer
Note: If the string value can not be converted, the value 0 is returned.		
Optional Value(s)	**Explanation**	