Documentation of the UJAC ExpressionInterpreter Type Handler set.

Handler for type byte[]

A type handler for byte[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type char[]

A type handler for char[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type double[]

A type handler for double[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type float[]

A type handler for float[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type int[]

A type handler for int[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type long[]

A type handler for long[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Object[]

A type handler for java.lang.Object[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type short[]

A type handler for short[] values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the array.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the array.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the array.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the array is empty or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the array is not empty.
size:	Determines the size of the array.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Boolean

A type handler for java.lang.Boolean values.

supported operations

!=:	Compares the object with the operand for inequality.
&&:	Combines the object and the operand with the 'and' operation.
==:	Compares the object with the operand for equality.
?:	Combines the object and the operand with the 'and' operation.
and:	Combines the object and the operand with the 'and' operation.
eq:	Compares the object with the operand for equality.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
ne:	Compares the object with the operand for inequality.
not:	Toggles the object's value: true becomes false and vice versa.
notDefined:	Checks if the object is not defined.
or:	Combines the object and the operand with the 'or' operation.
toString:	Serves a textual representation of the object.
||:	Combines the object and the operand with the 'or' operation.

Handler for type java.lang.Byte

A type handler for java.lang.Byte values.

supported operations

!=:	Compares the object with the operand for inequality.
%:	Calculates the modulo of the object and the operand value.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mod:	Calculates the modulo of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Character

A type handler for java.lang.Character values.

supported operations

!=:	Compares the object with the operand for inequality.
+:	Calculates the sum of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
concat:	Calculates the sum of the object and the operand value.
eq:	Compares the object with the operand for equality.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Double

A type handler for java.lang.Double values.

supported operations

!=:	Compares the object with the operand for inequality.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Float

A type handler for java.lang.Float values.

supported operations

!=:	Compares the object with the operand for inequality.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Integer

A type handler for java.lang.Integer values.

supported operations

!=:	Compares the object with the operand for inequality.
%:	Calculates the modulo of the object and the operand value.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mod:	Calculates the modulo of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Long

A type handler for java.lang.Long values.

supported operations

!=:	Compares the object with the operand for inequality.
%:	Calculates the modulo of the object and the operand value.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mod:	Calculates the modulo of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Object

A type handler for java.lang.Object values.

supported operations

!=:	Compares the object with the operand for inequality.
->:	Accesses the object's property specified by the operand.
.:	Accesses the object's property specified by the operand.
==:	Compares the object with the operand for equality.
[]:	Accesses the object's property specified by the operand.
eq:	Compares the object with the operand for equality.
getField:	Accesses the object's property specified by the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
propertyDefined:	Checks it the object contains a property that matches the name, specified by the operand.
toString:	Serves a textual representation of the object.

Handler for type java.lang.Short

A type handler for java.lang.Short values.

supported operations

!=:	Compares the object with the operand for inequality.
%:	Calculates the modulo of the object and the operand value.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mod:	Calculates the modulo of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.lang.String

A type handler for java.lang.String values.

supported operations

!=:	Compares the object with the operand for inequality.
+:	Concatenates the object and the operand.
==:	Compares the object with the operand for equality.
capitalize:	Raises the first character of the given String.
concat:	Concatenates the object and the operand.
eq:	Compares the object with the operand for equality.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Checks if the String has zero length.
length:	Gets the length of the object.
like:	Sorry, not documented yet.
lowerCase:	Gets the object as lower case string.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the String has no zero length.
notLike:	Compares the object with the operand for similarity. This operation behaves similar to the 'like' operation, known from SQL. In case the operand starts and ends with '%', text within the wildcard marks has to occur somewhere within the object. In case the operand only starts with '%', the object has to end with the operand text. In case the operand only ends with '%', the object has to start with the operand text. If none of the cases above matches, this operation behaves like the 'equal' operation.
substring:	Returns a substring of the object the operation is applied to. The operand can be given in the forms '<start>', '<start>:<length>', ':length'.
toString:	Serves a textual representation of the object.
trim:	Trims the leading and trailing white spaces from the object.
upperCase:	Gets the object as upper case string.

Handler for type java.math.BigDecimal

A type handler for java.math.BigDecimal values.

supported operations

!=:	Compares the object with the operand for inequality.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand.
ge:	Checks if the object is greater than or equal to the operand.
gt:	Checks if the object is greater than the operand.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
sub:	Calculates the difference of the object and the operand value.
toString:	Serves a textual representation of the object.

Handler for type java.sql.Time

A type handler for java.sql.Time values.

supported operations

!=: Compares the object with the operand for inequality.

->: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

.: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

<: Checks if the object is smaller than the operand.

<=: Checks if the object is smaller than or equal to the operand.

==: Compares the object with the operand for equality.

>: Checks if the object is greater than the operand.

>=: Checks if the object is greater than or equal to the operand.

beginOfYear: Gets the begin of year (January, 1st) for the date, specified by the object.

decrDay: Decrements the value of the object's day by the amount, specified by the operand.

decrMonth: Decrements the value of the object's month by the amount, specified by the operand.

decrYear: Decrements the value of the object's year by the amount, specified by the operand.

endOfYear: Gets the end of year (December, 31st) for the date, specified by the object.

eq: Compares the object with the operand for equality.

format: Formats the object using the format, specified by the operand.

ge: Checks if the object is greater than or equal to the operand.

getDay: Gets the day field from the date, specified by the object.

getMonth: Gets the month field from the date, specified by the object.

getYear: Gets the year field from the date, specified by the object.

gt: Checks if the object is greater than the operand.

incrDay: Increments the value of the object's day by the amount, specified by the operand.

incrMonth: Increments the value of the object's month by the amount, specified by the operand.

incrYear: Increments the value of the object's year by the amount, specified by the operand.

instanceof: Compares the object with the operand for inequality.

isDefined: Tells whether the object is defined or not.

le: Checks if the object is smaller than or equal to the operand.

lt: Checks if the object is smaller than the operand.

max: Gets the maximum of the object and the operand value.

min: Gets the minimum of the object and the operand value.

ne: Compares the object with the operand for inequality.

nextUltimo: Gets the next ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 30th of April.

notDefined: Checks if the object is not defined.

prevUltimo: Gets the previous ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 31th of March.

setDay: Sets the value of the object's day to the value, specified by the operand.

setMonth: Sets the value of the object's month to the value, specified by the operand.

setYear: Sets the value of the object's year to the value, specified by the operand.

toString: Serves a textual representation of the object.

Handler for type java.sql.Timestamp

A type handler for java.sql.Timestamp values.

supported operations

!=: Compares the object with the operand for inequality.

->: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

.: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

<: Checks if the object is smaller than the operand.

<=: Checks if the object is smaller than or equal to the operand.

==: Compares the object with the operand for equality.

>: Checks if the object is greater than the operand.

>=: Checks if the object is greater than or equal to the operand.

beginOfYear: Gets the begin of year (January, 1st) for the date, specified by the object.

decrDay: Decrements the value of the object's day by the amount, specified by the operand.

decrMonth: Decrements the value of the object's month by the amount, specified by the operand.

decrYear: Decrements the value of the object's year by the amount, specified by the operand.

endOfYear: Gets the end of year (December, 31st) for the date, specified by the object.

eq: Compares the object with the operand for equality.

format: Formats the object using the format, specified by the operand.

ge: Checks if the object is greater than or equal to the operand.

getDay: Gets the day field from the date, specified by the object.

getMonth: Gets the month field from the date, specified by the object.

getYear: Gets the year field from the date, specified by the object.

gt: Checks if the object is greater than the operand.

incrDay: Increments the value of the object's day by the amount, specified by the operand.

incrMonth: Increments the value of the object's month by the amount, specified by the operand.

incrYear: Increments the value of the object's year by the amount, specified by the operand.

instanceof: Compares the object with the operand for inequality.

isDefined: Tells whether the object is defined or not.

le: Checks if the object is smaller than or equal to the operand.

lt: Checks if the object is smaller than the operand.

max: Gets the maximum of the object and the operand value.

min: Gets the minimum of the object and the operand value.

ne: Compares the object with the operand for inequality.

nextUltimo: Gets the next ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 30th of April.

notDefined: Checks if the object is not defined.

prevUltimo: Gets the previous ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 31th of March.

setDay: Sets the value of the object's day to the value, specified by the operand.

setMonth: Sets the value of the object's month to the value, specified by the operand.

setYear: Sets the value of the object's year to the value, specified by the operand.

toString: Serves a textual representation of the object.

Handler for type java.util.Collection

A type handler for java.util.Collection values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the collection. This operation is only implemented for List instances
==:	Compares the object with the operand for equality.
[]:	Gets an element from the collection. This operation is only implemented for List instances
contains:	Checks if the collection contains the operand value.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the collection. This operation is only implemented for List instances
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the collection is empty or not.
ne:	Compares the object with the operand for inequality.
notContains:	Checks if the collection not contains the operand value.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the collection is not empty.
size:	Determines the size of the collection.
sort:	Sorts the collection. If an operand is given, it sorts the collection according to the bean field name the operand specifies.
toString:	Serves a textual representation of the object.

Handler for type java.util.Date

A type handler for java.util.Date values.

supported operations

!=: Compares the object with the operand for inequality.

->: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

.: Accesses the fields of the date by the given operand name.
Valid names are:
year: Get the year field of the date
month: Get the month field of the date
day: Get the day field of the date
hour: Get the hour of day field of the date
minute: Get the minute field of the date
second: Get the second field of the date
millisecond: Get the millisecond field of the date

<: Checks if the object is smaller than the operand.

<=: Checks if the object is smaller than or equal to the operand.

==: Compares the object with the operand for equality.

>: Checks if the object is greater than the operand.

>=: Checks if the object is greater than or equal to the operand.

beginOfYear: Gets the begin of year (January, 1st) for the date, specified by the object.

decrDay: Decrements the value of the object's day by the amount, specified by the operand.

decrMonth: Decrements the value of the object's month by the amount, specified by the operand.

decrYear: Decrements the value of the object's year by the amount, specified by the operand.

endOfYear: Gets the end of year (December, 31st) for the date, specified by the object.

eq: Compares the object with the operand for equality.

format: Formats the object using the format, specified by the operand.

ge: Checks if the object is greater than or equal to the operand.

getDay: Gets the day field from the date, specified by the object.

getMonth: Gets the month field from the date, specified by the object.

getYear: Gets the year field from the date, specified by the object.

gt: Checks if the object is greater than the operand.

incrDay: Increments the value of the object's day by the amount, specified by the operand.

incrMonth: Increments the value of the object's month by the amount, specified by the operand.

incrYear: Increments the value of the object's year by the amount, specified by the operand.

instanceof: Compares the object with the operand for inequality.

isDefined: Tells whether the object is defined or not.

le: Checks if the object is smaller than or equal to the operand.

lt: Checks if the object is smaller than the operand.

max: Gets the maximum of the object and the operand value.

min: Gets the minimum of the object and the operand value.

ne: Compares the object with the operand for inequality.

nextUltimo: Gets the next ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 30th of April.

notDefined: Checks if the object is not defined.

prevUltimo: Gets the previous ultimo of the date, specified by the object. The ultimo date is the last day of a month. In case the day is the 15th of April, the resulting object will be the 31th of March.

setDay: Sets the value of the object's day to the value, specified by the operand.

setMonth: Sets the value of the object's month to the value, specified by the operand.

setYear: Sets the value of the object's year to the value, specified by the operand.

toString: Serves a textual representation of the object.

Handler for type java.util.Map

A type handler for java.util.Map values.

supported operations

!=: Compares the object with the operand for inequality.

.: Gets an element from the map by its key.

==: Compares the object with the operand for equality.

[]: Gets an element from the map by its key.

containsKey: Checks if the map contains the key, specified by the operand value.

entries: Gets an element from the map by its key.

eq: Compares the object with the operand for equality.

get: Gets an element from the map by its key.

instanceof: Compares the object with the operand for inequality.

isDefined: Tells whether the object is defined or not.

isEmpty: Tells whether the map is empty or not.

ne: Compares the object with the operand for inequality.

notContainsKey: Checks if the map not contains the key, specified by the operand.

notDefined: Checks if the object is not defined.

notEmpty: Checks if the map is not empty.

size: Determines the size of the map.

toString: Serves a textual representation of the object.

Handler for type java.util.Map$Entry

A type handler for java.util.Map$Entry values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Accesses the object's key/value properties.
==:	Compares the object with the operand for equality.
[]:	Accesses the object's key/value properties.
eq:	Compares the object with the operand for equality.
get:	Accesses the object's key/value properties.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
toString:	Serves a textual representation of the object.

Handler for type java.util.ResourceBundle

A type handler for java.util.ResourceBundle values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Gets an element from the bundle by its name.
==:	Compares the object with the operand for equality.
[]:	Gets an element from the bundle by its name.
containsKey:	Checks if the bundle contains the key, specified by the operand value.
eq:	Compares the object with the operand for equality.
get:	Gets an element from the bundle by its name.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the bundle is empty or not.
ne:	Compares the object with the operand for inequality.
notContainsKey:	Checks if the bundle not contains the key, specified by the operand.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the bundle is not empty.
toString:	Serves a textual representation of the object.

Handler for type org.ujac.util.exi.ConditionResultHolder

A type handler for org.ujac.util.exi.ConditionResultHolder values.

supported operations

!=:	Compares the object with the operand for inequality.
::	Tells whether the object is defined or not.
==:	Compares the object with the operand for equality.
eq:	Compares the object with the operand for equality.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
ne:	Compares the object with the operand for inequality.
notDefined:	Checks if the object is not defined.
toString:	Serves a textual representation of the object.

Handler for type null Reference

A special handler for null values to avoid some error messages, resulting from not declared operations at the default handler.

supported operations

!=:	Compares the object with the operand for inequality.
****:***	Calculates the product of the object and the operand value.
+:	Calculates the sum of the object and the operand value.
-:	Calculates the difference of the object and the operand value.
->:	Accesses the object's property specified by the operand.
.:	Accesses the object's property specified by the operand.
/:	Calculates the quotient of the object and the operand value.
<:	Checks if the object is smaller than the operand.
<=:	Checks if the object is smaller than or equal to the operand.
==:	Compares the object with the operand for equality.
>:	Checks if the object is greater than the operand.
>=:	Checks if the object is greater than or equal to the operand.
[]:	Accesses the object's property specified by the operand.
abs:	Determines the object's absolute value.
add:	Calculates the sum of the object and the operand value.
div:	Calculates the quotient of the object and the operand value.
eq:	Compares the object with the operand for equality.
format:	Formats the object using the format, specified by the operand (always '').
ge:	Checks if the object is greater than or equal to the operand.
get:	Accesses the object's property specified by the operand.
getField:	Accesses the object's property specified by the operand.
gt:	Checks if the object is greater than the operand.
hasRows:	Checks if the object is not empty.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Tells whether the object is empty or not.
le:	Checks if the object is smaller than or equal to the operand.
lt:	Checks if the object is smaller than the operand.
max:	Gets the maximum of the object and the operand value.
min:	Gets the minimum of the object and the operand value.
mul:	Calculates the product of the object and the operand value.
ne:	Compares the object with the operand for inequality.
noRows:	Tells whether the object is empty or not.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the object is not empty.
propertyDefined:	Checks it the object contains a property that matches the name, specified by the operand.
rowCount:	Determines the size of the object (always 0).

size: Determines the size of the object (always 0).

sub: Calculates the difference of the object and the operand value.

toString: Serves a textual representation of the object.

Handler for type org.ujac.util.exi.SequenceIndex

A type handler for org.ujac.util.exi.SequenceIndex values.

supported operations

!=: Compares the object with the operand for inequality.

%: Calculates the modulo of the object and the operand value.

****:*** Calculates the product of the object and the operand value.

+: Calculates the sum of the object and the operand value.

-: Calculates the difference of the object and the operand value.

/: Calculates the quotient of the object and the operand value.

<: Checks if the object is smaller than the operand.

<=: Checks if the object is smaller than or equal to the operand.

==: Compares the object with the operand for equality.

>: Checks if the object is greater than the operand.

>=: Checks if the object is greater than or equal to the operand.

abs: Determines the object's absolute value.

add: Calculates the sum of the object and the operand value.

div: Calculates the quotient of the object and the operand value.

eq: Compares the object with the operand for equality.

format: Formats the object using the format, specified by the operand.

ge: Checks if the object is greater than or equal to the operand.

gt: Checks if the object is greater than the operand.

instanceof: Compares the object with the operand for inequality.

isDefined: Tells whether the object is defined or not.

le: Checks if the object is smaller than or equal to the operand.

lt: Checks if the object is smaller than the operand.

max: Gets the maximum of the object and the operand value.

min: Gets the minimum of the object and the operand value.

mod: Calculates the modulo of the object and the operand value.

mul: Calculates the product of the object and the operand value.

ne: Compares the object with the operand for inequality.

notDefined: Checks if the object is not defined.

sub: Calculates the difference of the object and the operand value.

toString: Serves a textual representation of the object.

Handler for type org.ujac.util.table.Row

A type handler for org.ujac.util.table.Row values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Accesses the field specified by the operand at this row.
==:	Compares the object with the operand for equality.
[]:	Accesses the field specified by the operand at this row.
columnCount:	Gets the number of columns from the table that holds the row.
eq:	Compares the object with the operand for equality.
formatValue:	Gets the formatted value of the field specified by the operand at this row.
getField:	Accesses the field specified by the operand at this row.
hasColumn:	Checks it the table that holds the row contains a column that matches the name, specified by the operand.
hasRows:	Checks if the table that holds the row, has rows.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isVisible:	Checks whether the field, specified by the operand is visible or not.
ne:	Compares the object with the operand for inequality.
noRows:	Checks if the table that holds the row, has no rows.
notDefined:	Checks if the object is not defined.
rowCount:	Gets the number of rows from the table that holds the row.
rowType:	Gets the type of the row.
toString:	Serves a textual representation of the object.

Handler for type org.ujac.util.table.Table

A type handler for org.ujac.util.table.Table values.

supported operations

!=:	Compares the object with the operand for inequality.
.:	Accesses the field specified by the operand at the first row of the table.
==:	Compares the object with the operand for equality.
[]:	Accesses the field specified by the operand at the first row of the table.
columnCount:	Gets the number of columns from the table, specified by the object.
columns:	Gets the number of columns from the table, specified by the object.
eq:	Compares the object with the operand for equality.
formatValue:	Gets the formatted value of the field specified by the operand at this row.
getField:	Accesses the field specified by the operand at the first row of the table.
getRow:	Accesses the field specified by the operand at the first row of the table.
hasColumn:	Checks it the table contains a column that matches the name, specified by the operand.
hasRows:	Checks if the table, specified by the object has rows.
instanceof:	Compares the object with the operand for inequality.
isDefined:	Tells whether the object is defined or not.
isEmpty:	Checks if the table, specified by the object has no rows.
ne:	Compares the object with the operand for inequality.
noRows:	Checks if the table, specified by the object has no rows.
notDefined:	Checks if the object is not defined.
notEmpty:	Checks if the table, specified by the object has rows.
rotate:	Rotates the table, specified by the object. It serves a table, that holds the same values as the original table, but rotated by 90 degrees. The first column of the rotated table holds the title row of the original table.
rowCount:	Gets the number of rows from the table, specified by the object.
size:	Gets the number of rows from the table, specified by the object.
toString:	Serves a textual representation of the object.
visibleColumns:	Gets the number of columns from the table, specified by the object.