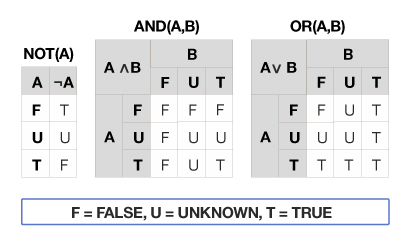
# Handling null values in IBM ODM

You can automate exception handling at the decision service project level so that the rule engine continues to process rule conditions that generate Java exceptions. When you activate automatic exception handling in conditions, the engine handles four Java exception types by default, but you can customize the list of Java exception types that the rule engine handles.

The rule engine ignores unknown values during the evaluation of the rule condition when automatic exception handling is enabled. You can enable and disable this feature in Rule Designer in the Rule Engine panel of the rule project properties window. Automatic exception handling is disabled by default.

When you enable automatic exception handling, The following subclasses are handled by the rule engine when automatic exception handling is enabled:

* ArithmeticException
* NumberFormatException
* NullPointerException
* IndexOutOfBoundsException

With automatic exception handling in conditions enabled, expressions in conditions that result in exceptions during their evaluation are treated as unknown values and are handled by the rule engine. The rule engine evaluates the overall rule conditions by using the following three-valued logic:  
  


If the rule engine evaluates the rule condition as true or false, despite unknown values in the condition expressions, the rule action logic is applied. If the rule engine evaluates the rule condition as unknown, because of unknown values in the condition expressions, the rule is not fired.

For example, in the following rule, when the value of the name of the customer is null, and the age of the customer is 20, the first condition cannot be resolved, and the second condition is met:

if

the name of the customer is 'Paul'

or the age of the customer is 20

then

reject the loan;

Because the rule uses the logical or operator, the action to reject the loan is taken. If the age of the customer is not 20, the first condition cannot be resolved, and the second condition is not met. In this case, the rule is not fired, and the loan is not rejected.

## **Conditions that use collections**

When the rule engine assembles [collections](https://www.ibm.com/docs/en/SSQP76_8.9.0/com.ibm.odm.dserver.rules.designer.author/config_auth_topics/con_rd_bom_collections.html) of objects, the objects for which the where clause is unknown or false are ignored. Consequently, the test on the number of objects in a collection reflects this behavior: Only objects that have a where clause evaluated true are counted.

The following business language expressions use tests on the number of objects:

* If the number of ...
* If there is no ...
* If there is at least one ...
* If there are at least <number> ...
* If there are more than <number> ...
* If there are less than <number> ...
* If there are <number> ...

The following example shows a condition that uses collections and tests the number of objects. In the example, the collection (the borrowers) contains only borrowers for whom the attribute values are known. So, when the count operation is performed (is less than 3), any borrower for whom an attribute is unknown is not included in the count. In other words, if 3 borrowers are evaluated and the name of one borrower is unknown while the name of the two other borrows is the name of the 'borrower', then the condition is True and the then action is taken.

if

the number of borrowers where the name of each borrower is the name of 'the borrower',

is less than 3,

then

add "less than 3 similar barrowers found" to the messages of 'the loan' ;

else

add "duplicate detected" to the messages of 'the loan' ;

reject 'the loan' ;

In the following example, if the where clause is true for two borrowers and unknown for two borrowers, then the rule engine considers that the rule condition is false. The else action is taken:

If there are more than 3 borrowers in the past borrowers of 'the loan' where the name of each borrower is the name of 'the borrower',

then

reject 'the loan';

add "more than 3 similar borrowers found" to the messages of 'the loan' ;

else

add "3 or fewer similar borrowers found " to the messages of 'the loan' ;

For more examples, see [Expressions that use collections](https://www.ibm.com/docs/en/SSQP76_8.9.0/com.ibm.odm.dserver.rules.designer.run/shared_optimizing/con_excpt_examples.html#concept_cjb_3yh_fy__bal).

## **Adding exception types**

You can customize the list of exception types automatically handled by creating a BOM with the corresponding classes and by adding specific properties, de.autoCatchExceptionInConditions and de.doNotAutoCatchExceptionInConditions, to each class where appropriate. When automatic exception handling is enabled in Rule Designer, the behavior is to handle automatically exceptions that occur in the condition parts of rules for exceptions that belong to the following subclasses:

* ArithmeticException
* NumberFormatException
* NullPointerException
* IndexOutOfBoundsException

You can customize the default behavior to add more exception types to be automatically handled by setting a property named de.autoCatchExceptionInConditions to the corresponding class using the BOM editor:

package java.lang;

public class ClassCastException

extends java.lang.RuntimeException

**property "de.autoCatchExceptionInConditions" "true"**

{

public ClassCastException();

public ClassCastException(string arg);

}

## **Excluding exception types**

You can exclude exception types from automatic exception handling in one of the handled classes by overriding the definition of the class in the BOM with the property named de.autoCatchExceptionInConditions. For example, you can remove arithmetic exceptions from automatic exception handling by removing the de.autoCatchExceptionInConditions property from the ArithmeticException class:

public class ArithmeticException

extends java.lang.RuntimeException

{

public ArithmeticException();

public ArithmeticException(string arg);

}

You can exclude a subclass of the four default classes from automatic exception handling by setting a property named de.doNotAutoCatchExceptionInConditions (in bold):

public class StringIndexOutOfBoundsException

extends java.lang.IndexOutOfBoundsException

**property "de.doNotAutoCatchExceptionInConditions" "true"**

{

public StringIndexOutOfBoundsException();

public StringIndexOutOfBoundsException(string arg);

public StringIndexOutOfBoundsException(int arg);

}

## **Logging**

Automatic exception handling uses the standard Java™ logger logging service, so that you can log messages based on message type and level and control how these messages are formatted and stored at runtime.

In Rule Designer, to capture activity traces that occur in automatic exception handling, you must first define a logging.properties file and make it available to the JVM that is used in the rule execution run or debug configuration. Then, in the Run Configurations window, under the Parameters & Arguments tab of your decision operation, enter one of the following VM arguments: -Djava.util.logging.config.file=file\_path/logging.properties or -Djava.util.logging.config.file="${workspace\_loc}/your\_project/logging.properties".

In the following example, the log level for the logger name com.ibm.rules.engine.aeh is set to FINE.

**handlers = java.util.logging.ConsoleHandler**

############################################################

# Handler specific properties.

# Describes specific configuration info for Handlers.

############################################################

# Limit the messages that are printed on the console to INFO and above.

java.util.logging.ConsoleHandler.level = FINE

java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter

**com.ibm.rules.engine.aeh.level = FINE**

To enable automatic exception handling logs in a server environment, define a logging.properties file and make it available to the JVM by configuring your application server, the log level for the Logger name com.ibm.rules.engine.aeh should be set to FINE.

For an introduction to the Java Logging API, see [Java Logging Overview](https://www.ibm.com/links?url=https%3A%2F%2Fdocs.oracle.com%2Fjavase%2F8%2Fdocs%2Ftechnotes%2Fguides%2Flogging%2Foverview.html). For more information about the java.util.logging Java logger, see [java.util.logging](https://www.ibm.com/links?url=https%3A%2F%2Fdocs.oracle.com%2Fjavase%2F7%2Fdocs%2Fapi%2Fjava%2Futil%2Flogging%2Fpackage-summary.html" \o "(Opens in a new tab or window)" \t "_blank) in the Oracle documentation.