



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

# GRS Best Practice Guide

Working with Composer's Business Rules Block

12/14/2025

# Working with Composer's Business Rules Block

## Contents

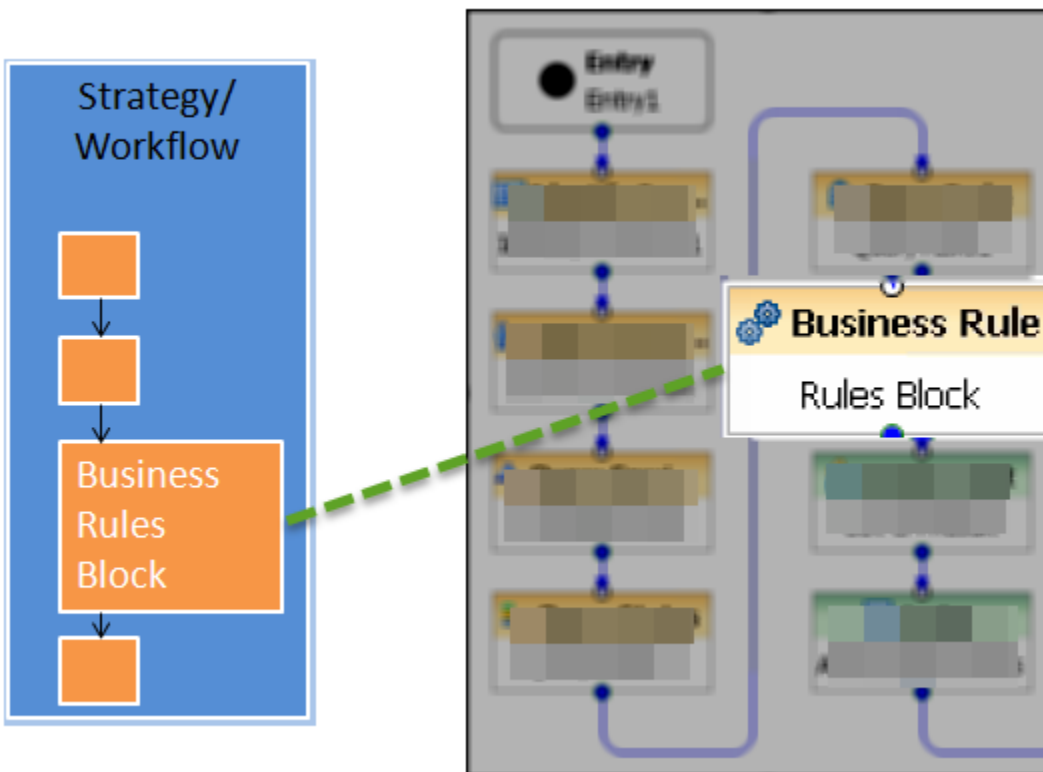
- [1 Working with Composer's Business Rules Block](#)
  - [1.1 Working with Composer Workflows](#)
  - [1.2 Business Rule Block Parameters and Properties](#)
  - [1.3 Working With Returned Data](#)

## Working with Composer Workflows

Once the Rule Package (created from Rule Templates) that you want to work with is deployed to the Genesys Rules Engine, you can use the Business Rule block on the Composer Server Side palette to create voice and routing applications that use business rules.

### Important

In the process flow diagram at the beginning of this document, we used a schematic to indicate how the Composer Business Rule block fits into a workflow. In the graphic below you can see that we are not concerned with the details of the workflow application, only in how a rule that a rule author has created is consumed by the Composer/ORS application that the workflow represents.



To edit the Business Rule block in Composer, double-click it.

## Important

Because you will specify a rules package and a rule engine in this block, you must have already deployed a rules package to the relevant rules engine and configured a connection to the rule engine before the Business Rule block can be completed.

## Business Rule Block Parameters and Properties

When you open the Business Rule block, it looks something like this:

| Model      | Property              | Value                          |
|------------|-----------------------|--------------------------------|
| Appearance | Alias                 |                                |
|            | Name                  | CallTreatmentRule              |
|            | Annotation            |                                |
|            | Block Notes           |                                |
|            | Business Rule         |                                |
|            | Business Rule Package |                                |
|            | Facts                 |                                |
|            | Rules Engine URL      |                                |
|            | Exceptions            |                                |
|            | Exceptions            |                                |
|            | Logging               |                                |
|            | Condition             |                                |
|            | Logging Details       |                                |
|            | Log Level             | Project Default: Error         |
|            | Output                |                                |
|            | Output Result         | Variable(callTreatmentResults) |

Use this block to have Composer query the Genesys Rules Authoring Tool (GRAT) for deployed packages. For the Rule Package that you specify, Composer will query the GRAT for the Facts associated with the Rule Package. You can then set values for the Facts, call the Genesys Rules Engine for evaluation, and save the results in a variable.

## Important

This last step (evaluation) happens as part of a strategy or workflow application that Composer developer creates.

The Business Rule block has, among others, the following properties:

## Business Rule

### Business Rule Package Property

Use to select the Rule Package (collection of related rules) you would like to execute. The rule package must have already been deployed to the GRE. You will only be able to select deployed rule packages.

Packaging rules together allows the business analyst to define which rules will support a particular application. Before using this property, you must request Composer to connect to the Genesys Rules Authoring Tool Server using the information specified in Business Rule Preferences. After a successful connection, the Business Rule Package dialog appears.

Select a Rule Package and click **OK**. The dialog closes and the name of the Rule Package appears under Value.

### Facts Property

Once you have selected the rule package, the Fact model associated with the template that the rule package is based on will automatically populate as selections in a drop-down menu in the Fact Class field.

Example:

| Name    | Data Type | Value |
|---------|-----------|-------|
| segment | string    |       |
| age     | integer   |       |
| gender  | string    |       |
|         |           |       |
|         |           |       |
|         |           |       |

Select a Fact class and create a new Fact based on it. This Fact will display all the fields that were defined in the template. At this point you can assign values to those Fact fields.

These values can be literals or variables that the workflow application is using. The available values are displayed in the Values drop-down menu.

You must also define here the output Fact (Output Result Property below)—the Fact that will contain the decision made by the rules engine and is passed back to the rule block in the workflow application to be processed by the workflow logic.

### Rules Engine URL

Select the variable containing the Genesys Rules Engine URL.

### Output Result Property

Use this property to save the results of the business rule execution to a variable.

The format of returned data is JSON. Any post-processing work to be done on returned results can be done in the existing Assign Block which provides access to ECMAScript functions. It supports writing simple or complex expressions to extract values out of JSON strings and arrays. In a workflow, the Output Result can be attached to User Data.

#### Important

The **Output Result** property takes effect only during application runtime. Its purpose is to take the output of the rule execution (at runtime) and store returned results back in the specified application variable so other parts of the application can access the data.

## Working With Returned Data

Below is an example (using output variable `callTreatmentResults`) on how to work with data returned by the Business Rules block. A sample of the output can look like the snippet below, which will be stored in `callTreatmentResults`.

```
callTreatmentResults='{
  "knowledgebase-response":{
    "InOutFacts":{"named-fact":[
      {
        "id":"customer",
        "fact":{
          "@class":"call.treatment.Customer",
          "segment":"gold"
        }
      },
      {
        "id":"callinfo",
        "fact":{
          "@class":"call.treatment.CallInfo",
          "intention":"Address Change"
        }
      }
    ]}
  }
}
```

```
    {
        "id": "callTreatment",
        "fact": {
            "routeToAgentGroup": "Accounts"
            "@class": "call.treatment.CallTreatment",
            "maxWaitTime": "10"
            "offerCallBack": "false"
        }
    }
  ]}
}}
```

This data structure contains the evaluation of the rule(s) as determined by the rules engine. These results are passed back to the Composer application for decision-making and onward processing.