Composer Help

Expression Builder

5/10/2025

Expression Builder

Contents

Use for both voice callflows and routing workflows to build expressions for branching and conditional routing decisions. You create expressions in Expression Builder; you can assign them to variables using the Assign block. You can also build ECMAScript expressions that use the Genesys Functional Modules in Expression Builder). Also see Skill Expression Builder used for routing applications.

Opening Expression Builder

Expression Builder opens from the various blocks, which include but are not limited to:

- Assign--Assign Data Property
- Branching--Conditions Property
- ECMAScript (for workflows)--Script Property
- Entry--Variables Property
- Log--Logging Details Property
- Looping--Exit Expression Property

Data Categories Accessed

Expression Builder gives access to the following categories of data, which can be used in expressions:

- Project Variables
- Workflow Variables (if accessed from a workflow)
- Callflow Variables (if accessed from a callflow)
- Workflow Functions (if accessed from a workflow)
- Callflow Functions (if accessed from a callflow)
- JavaScript (Array, Boolean, Date, Math, Number, String functions)
- Orchestration Server Functions
- Context Services
- Configuration Server

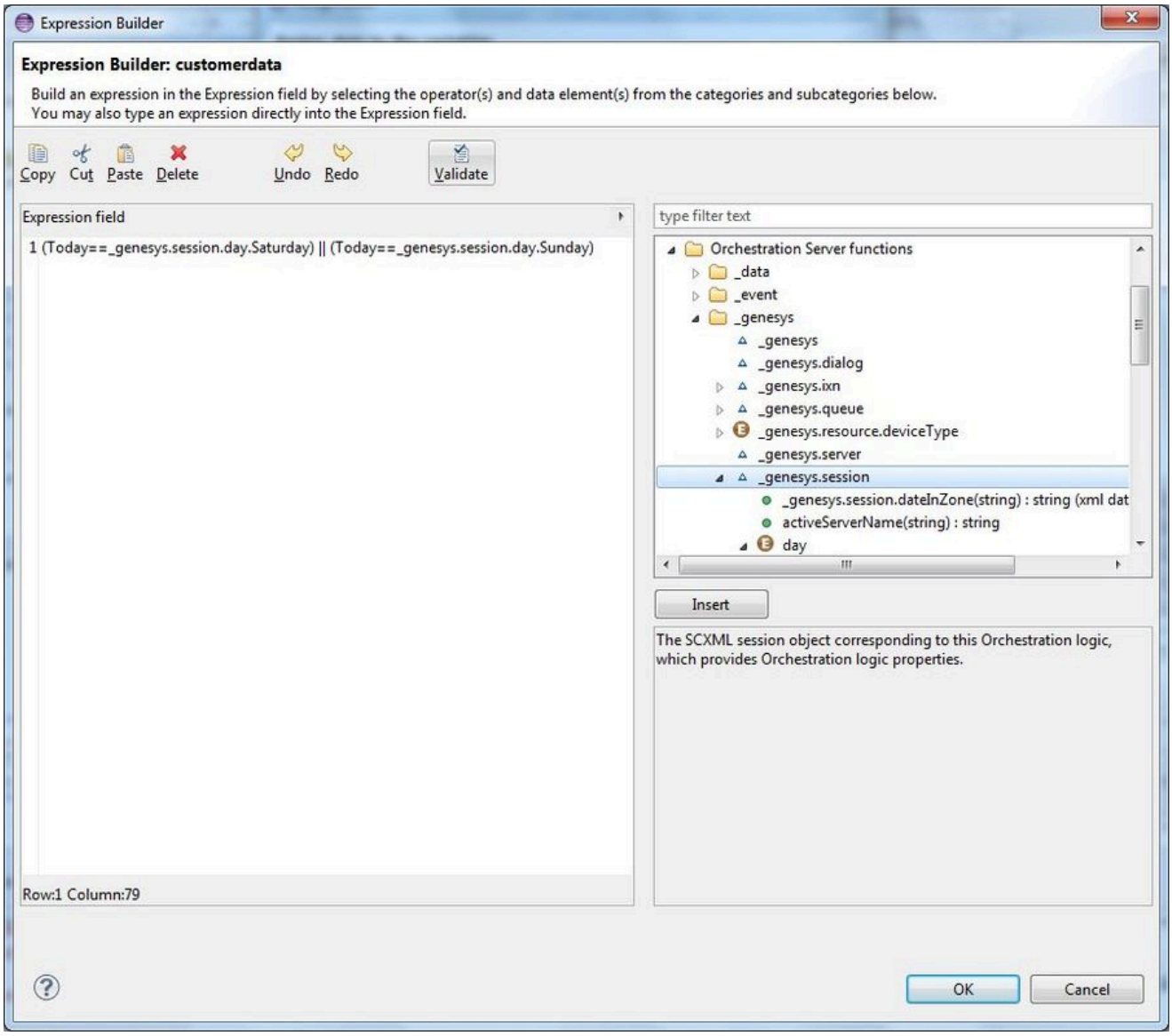
Note: Depending on the calling context (IPD, workflow editing, callflow editing), some of the above categories may be hidden.

Expression Builder Toolbars

This section discusses the editing toolbar the top of Expression Builder as well as the Operators toolbar.

Editing Toolbar

Expression Builder has an editing toolbar with buttons representing copy, cut, paste, delete, undo, redo, and validate . After you entering an expression and click the button to validate, syntax messages appear under the Expression Builder title. In the figure below, the syntax message is: No syntax error was found.



Operators

Expressions can consist of comparisons joined by AND (&) and OR (|), which have the same priority. URS uses integer arithmetic in its calculations, such as for expression evaluation. For this reason, you must always create expressions based on integer arithmetic, not float. When an expression contains more than one logical comparison, the logical operation to the left has precedence over the operation

to the right. Use parentheses to overrule this precedence. When the order of logical operations is not explicitly specified by parentheses, the operation to the left has precedence over the operation to the right. For example: `Portuguese > 5 | Africa = 7 & SpTours >3` is equivalent to `(Portuguese > 5 | Africa = 7) & SpTours > 3` Each comparison consists of two data values that are compared against each other. The table below shows the operators used in logical expressions.

Symbol	Meaning	Example
==	use for comparison (e.g. x==y to compare is x equal to y)	see example screen shot
=	equal to done for assignment (e.g. x=y to assign x equal to y)	Day = Monday
!=	not equal to	Day != Sunday
&	Single & is an AND operators which manipulates internal bits of integers	
&&	&& and || are used for comparisons to build boolean (true/false) expressions	
||	see &&	see example screen shot
>	greater than	Time > 9:00
>=	greater than or equal to	Day >= Monday
<	less than	Time < 16:00
<=	less than or equal to	Day <= Friday

Mathematical Symbols Allowed

The allowed mathematical symbols in expressions are restricted to those symbols for addition, subtraction, division, and multiplication (+, -, /, *). These symbols work with numeric values only, so any participating String argument is automatically converted to a number.

Examples of Expressions

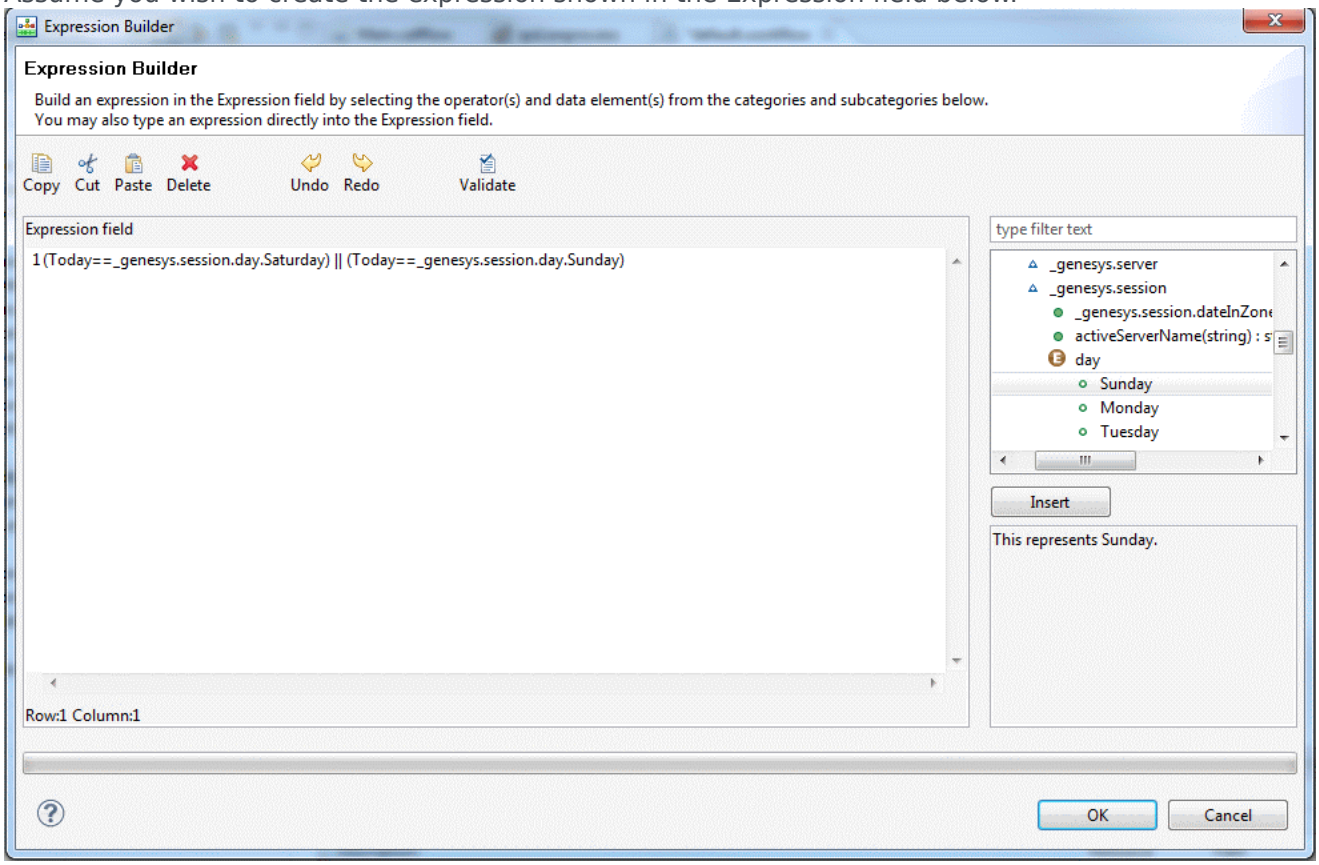
The table below shows example expressions.

Example	Interpretation
Day=Saturday|Day=Sunday	If the day is Saturday or Sunday
Time<=5:00 |Time>=18:00	If the time is less than or equal to 5:00 (5:00 AM) or greater than or equal to 18:00 (6:00 PM); in other words, if the time is between 6:00 PM and 5:00 AM.
Day>=Monday&Day<=Friday & ANI!=8004154732	If the day is a weekday and the ANI is not 8004154732
SData[1123@SF.A,StatTimeInReadyState]>120	If agent 1123 has been in Ready state for more than 120 seconds

UData[Acct]>=9000	If the value associated with the key Acct in the Interaction data structure is equal to or greater than 9000

Creating Expressions

Assume you wish to create the expression shown in the Expression field below.



In the above figure, the expression `(Today ==_genesys.session.day.Saturday) || (Today == _genesys.session.day.Sunday)` was created as follows:

- In the left text field under the toolbar, type an open parenthesis (().
- If you already defined the "Today" variable, select it under **Workflow variables** or **Project variables**. Otherwise, if not yet defined, type "Today" (no quotes) in the text after the open parenthesis.
- Type "==" (no quotes) in the text.
- In the right text box, expand:

`Orchestration Server Functions _genesys genesys.session.day`

- Double-click **Saturday**.
- In the left text field, type a close parenthesis ())

- Click the **|| Operators** button. Repeat the above steps except, at the end before the close parenthesis, select _genesys.session.day.Sunday.

Note: You may also use the search field on the right side to filter items that include **Saturday**

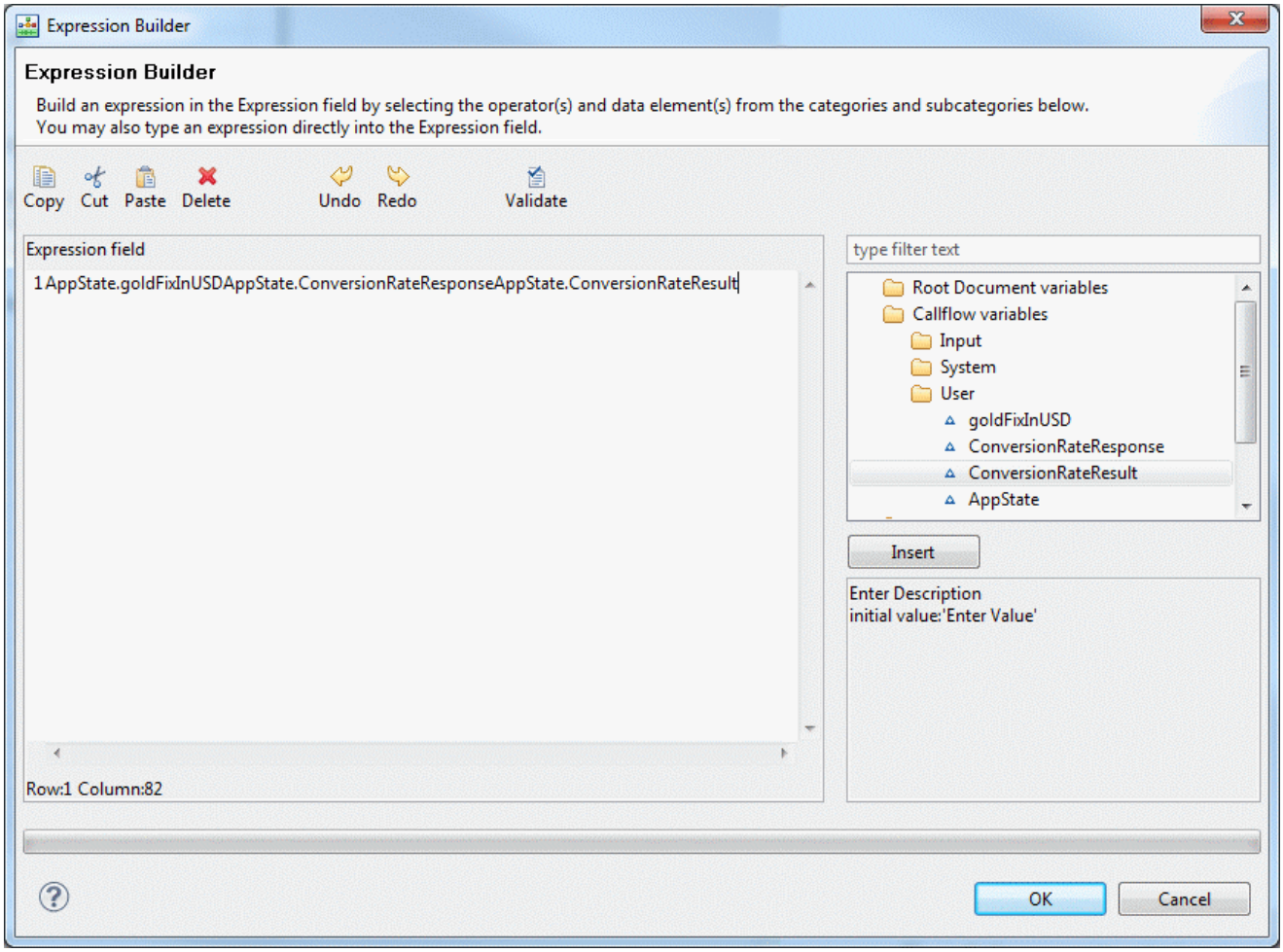
Usage of Variables in Expressions

Note: The steps for using variables in Expression Builder varies slightly depends on whether you are creating a voice callflow or a routing workflow.

- If creating a voice callflow, the right selection area contains Callflow Variables.
- If creating a routing workflow, the selection area contains Workflow Variables.

Using Variables in a Callflow

Assume you wish to use variables in a callflow to create the following: `AppState.goldFixInUSDAppState.ConversionRateResponseAppState.ConversionRateResult` The figure below shows the entry in Expression Builder.



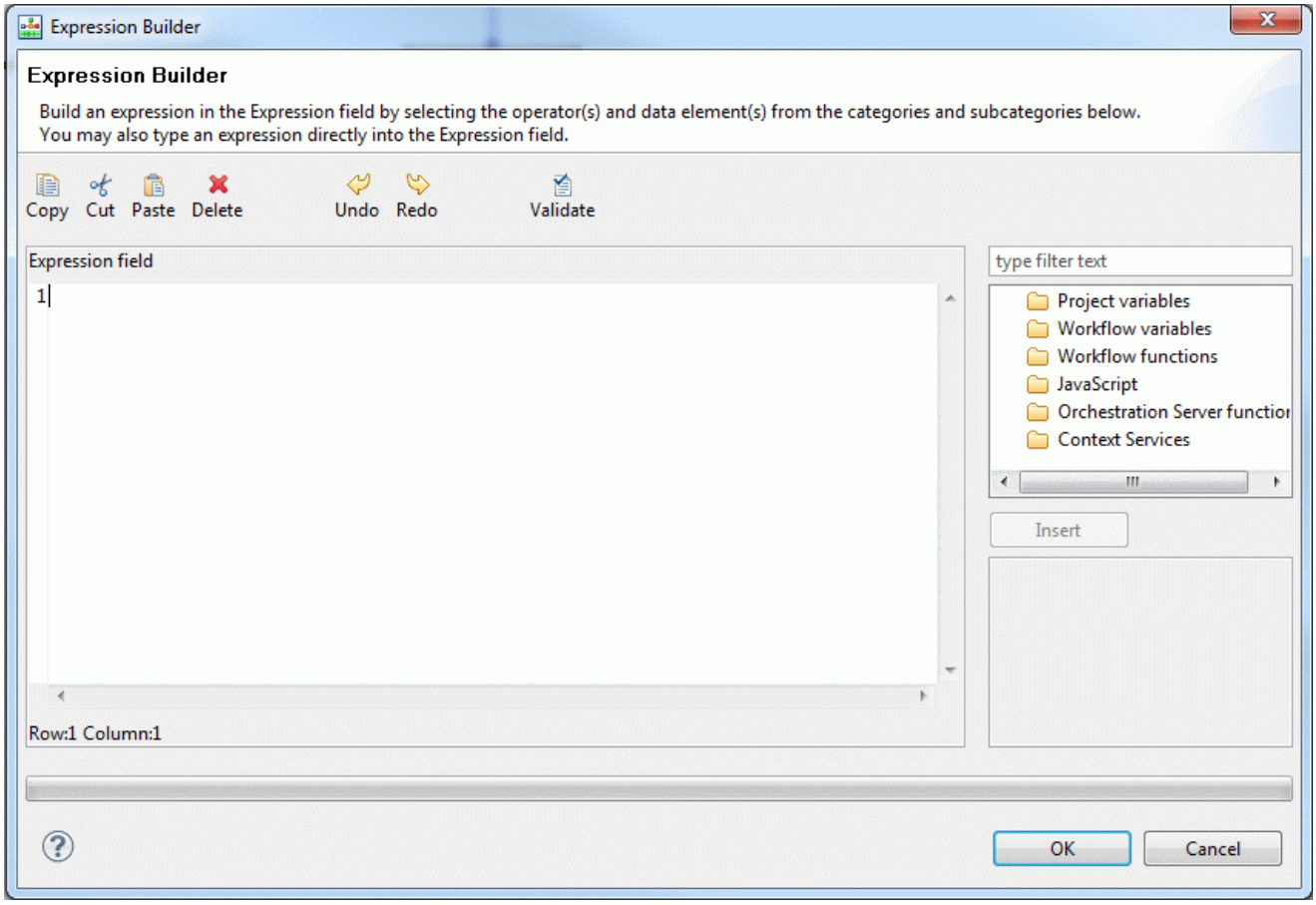
Assume you have already defined these variables in the Entry object. The above entry could then be created as follows:

1. The right selection area lists **Callflow Variables**, **Java Script**, and **Context Services**.
2. Expand **Callflow Variables**.
3. Expand **User** to view variables defined in the Entry block.
4. To place the variables in the Expression field at the top of Expression Builder, double-click **goldFixInUSD**,**ConversionRateResponse**, and **ConversionRateResult**. The AppState VoiceXML Data Model Object will be appended automatically to variables used inside Expression Builder. The code inside the Expression field will be directly substituted within the generated VoiceXML code.
5. Continue creating the expression.

Expression Builder Data Categories

Expression Builder accesses various categories of data Including Orchestration Server and Universal

Routing Server functions. The folders shown on the right depend on whether you are working with a callflow or workflow. The figure below shows Expression Builder Data Categories when working with a workflow.



Project variables

Use Project variables when you need to share information across different workflows.

Workflow variables

Use Workflow variables when you need to share information across different blocks in the same workflow.

Workflow functions

Use the Workflow functions category when creating routing workflows. Selecting a function displays a description. See the Orchestration Server wiki for information on functions available for use in Composer when building routing workflows. **Notes**: Functions getCallType and getIxnMediaType can be used to identify the call type and/or media type for the purpose of segmenting incoming interactions.

- `getCallType(ixnID)`--This function gets the call type `_genesys.ixn.interactions[].voice.type` for the specified interaction. If the ixnID is not specified, it will return the calltype for the current interaction.
- `getIxnMediaType(ixnID)`--This function gets the correct media type ENUM from `_genesys.FMName.MediaType` for the specified ixnID. If ixnID is not specified, the current interaction id will be used. If the interaction's media type cannot be determined or the specified ixnID does not exist, the function will return undefined.

Use Case: A call arrives on a routing pointing, initiating a routing workflow. The workflow checks for the call type. If the call type is outbound, then the call is immediately moved to the front of the queue and routed to an available agent. If the call type is inbound, the call is assigned a priority and routed based on the desired service level. For information on the other functions, see eServices Blocks.

Callflow functions

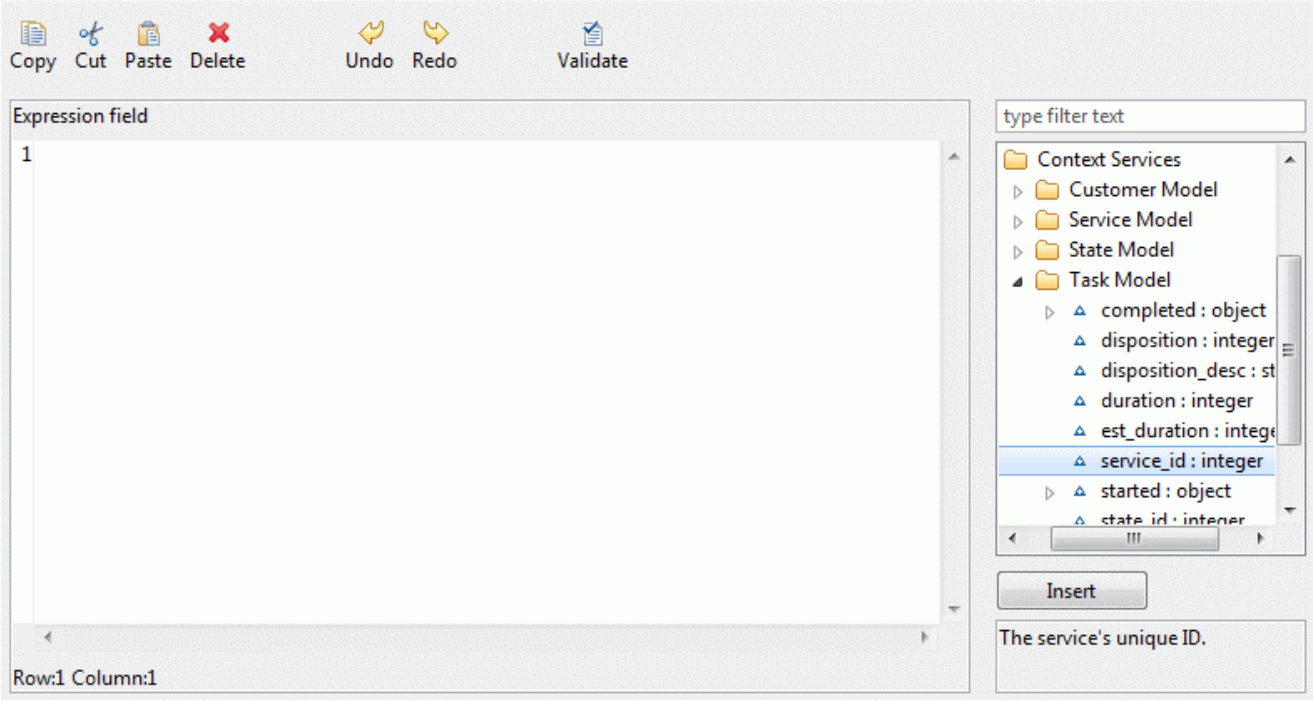
Note: Function `getSIPHeaderValue(headername)` returns the SIP header value associated with the given SIP headername. You may wish to use this function with the Assign block.

JavaScript

Use JavaScript to access those functions categorized as follows: Array, Boolean, Date, Math, Number, String.

Context Services

Use Context Services when creating expressions that use attributes associated with this optional capability of Universal Contact Server Database.



Configuration Server

This category is displayed for workflows when the Expression Builder is called from the ECMAScript and Branching blocks. If connected to Configuration Server, Composer can fetch standard responses and category codes.

Standard Response Library

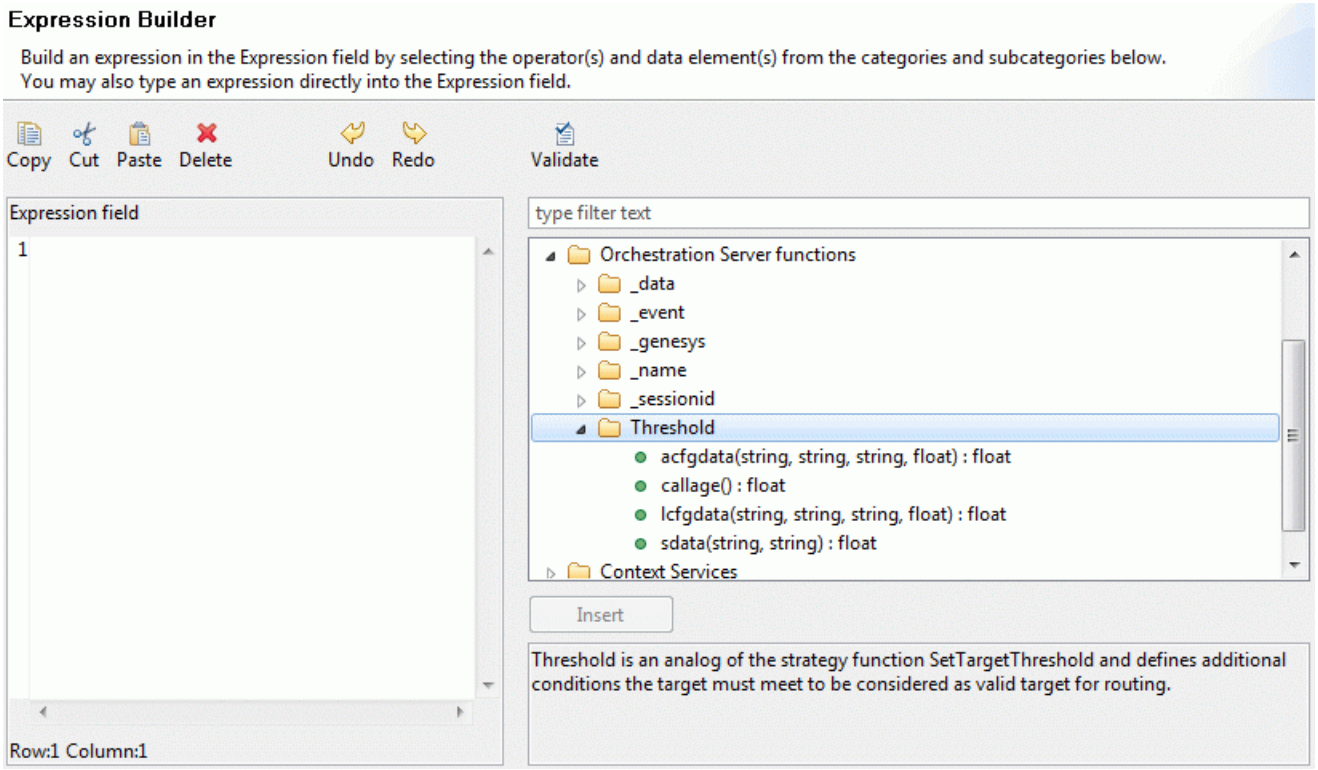
This category allows you to access pre-written responses for customers that have been defined in Knowledge Manager (as described in the *eServices 8.1 User's Guide*).

Orchestration Server and URS Functions

Composer's Expression Builder provides access to many Orchestration Server and Universal Routing Server (URS) functions. For more information see Using URS Functions.

Threshold Functions

Universal Routing Server's threshold functions can be used for conditional routing. For example, the threshold functions can be used in the Target block for a type of conditional routing called "share agents by service level agreement routing." This type of routing enables a business user that manages multiple business lines to define the triggering conditions and constraints that allow agents to be shared among business lines. By constructing a single threshold expression, you can define the triggering conditions for borrowing agents from other business lines as well as the conditions that apply to the lending business line.
Threshold is an analog of the URS strategy function `SetTargetThreshold` as defined in *Universal Routing 8.1 Routing Application Configuration Guide*. It defines additional conditions the target must meet to be considered as valid target for routing. The threshold functions are available in Composer's Expression Builder:



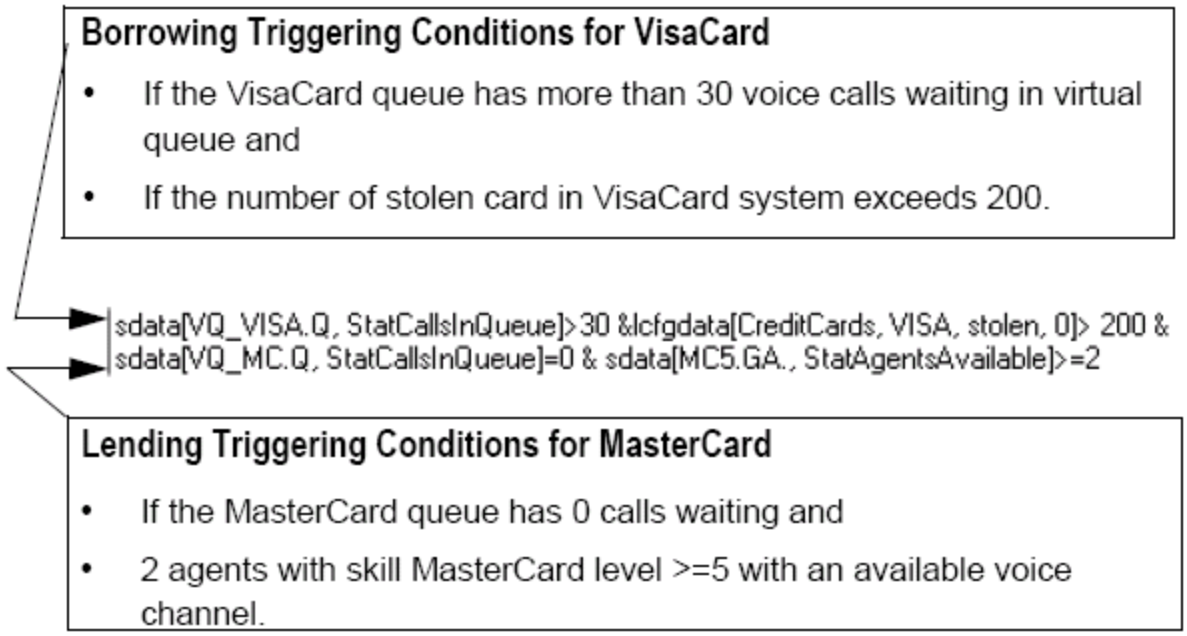
- `acfgdata(Application name, folder, property, default value)`. Use this function to affect routing conditions based on external data stored in properties of Configuration Database Application objects (ApplicationConFigDATA). Returns a numeric value for a specified Application option. If an Application has no such option then the default value is returned. Return value type: FLOAT. Example: sdata(Group2.GA, StatAgentsAvailable)>acfdata(URS, default, MinNumOfRdyAgents, 2)
- `callage`. Use this function to return the age of an interaction in seconds. Use for time-based routing conditions, such as a call that can only be routed if it waits more then 60 seconds. Return value type: FLOAT.
- `lcfgdata(list name, item, attribute, default value)`. Use this function to affect routing conditions based on external data stored in List objects. Returns a numeric value for a specified attribute of a List object's item If a List object has no such item or attribute, the default value is returned. Works like acfgdata, but uses a List object (`ListConFigDATA`) instead of an Application. Return value type: FLOAT.
- `sdata(target, statistic)`. Use this function to affect routing conditions based on statistics. Specify targets and statistics just like for the `SData[]` function described in the *Universal Routing 8.1 Reference Manual*. You can use the URS predefined statistics (see Statistics Manager and Builder), such as: PositionInQueue, CallsWaiting, and InVQWaitTime. Return value type: FLOAT. Example: sdata(Group2.GA, StatAgentsAvailable)>2

Example Threshold Expression

A threshold expression is text string very similar to the regular expressions used for branching, but uses the threshold functions. In the example below, `sdata` and `lcfgdata` are the predefined threshold functions.

```
sdata[VQ_VISA.Q, StatCallsInQueue]>30 &lcfgdata[CreditCards, VISA, stolen, 0]> 200 &
sdata[VQ_MC.Q, StatCallsInQueue]=0 & sdata[MC5.GA., StatAgentsAvailable]>=2
```

In this example, both the borrowing and lending conditions are defined in a single threshold expression:



For detailed information on using the threshold functions, see *Universal Routing 8.1 Routing Application Configuration Guide* and threshold attribute in Orchestration Developers Guide.

GetMediaTypeName Function

This function returns the name of media type associated with interaction as defined in the Configuration Database. Located in Expression Builder as follows: `_genesys.ixn.mediaType` Expression Builder lists the Media type enumerators supported by the URS platform. The main difference between this function and IRD's `getMediaTypeName` function is that

- The IRD function takes a parameter for (the current interaction media type) and returns a String name of the media type

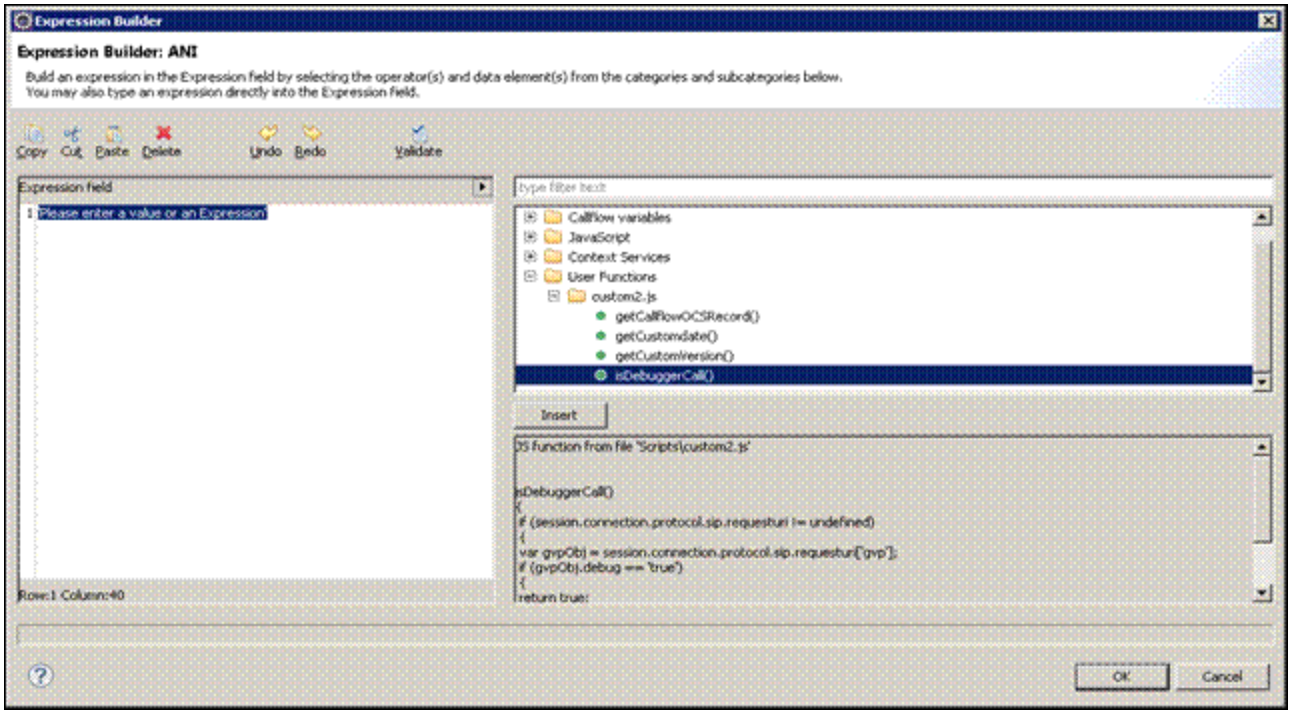
Whereas:

- Composer's function `getIxnMediaType(ixnID)` takes a parameter of any interaction and returns a ENUM type `_genesys.FMName.MediaType` of the media type. Returns the media type of the given interaction (ixnID), otherwise the current interaction.

User defined JavaScript Functions

Expression Builder shows the user defined JS methods under 'User Functions' category to list user defined JS methods for easy access.

- For Callflow diargams, JavaScript files added in the Scripts property of the Entry BlockScripts are considered.
- For Workflow diargams, JavaScript files added inside the include/user folder are considered.


Expression Builder
Expression Builder: ANI
Build an expression in the Expression field by selecting the operator(s) and data element(s) from the categories and subcategories below.
You may also type an expression directly into the Expression field.
Copy
Cut
Paste
Delete
Undo
Redo
Validate
Expression field
Please enter a value or an Expression
Row:1 Column:40
type filter text
Callflow variables
JavaScript
Context Services
User Functions
custom2.js
getCallflowOCSRecord()
getCustomDate()
getCustomVersion()
isDebuggerCall()
Insert
JS function from file 'Scripts\custom2.js'
isDebuggerCall()
{
if (session.connection.protocol.sip.requesturi != undefined)
{
var gvpObj = session.connection.protocol.sip.requesturi['gvp'];
if (gvpObj.debug == 'true')
{
return true;
OK
Cancel