



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.

Composer Help

Skill Expression Builder

Skill Expression Builder

Contents





- **1 Skill Expression Builder**
 - 1.1 Using Skill Expression Builder
 - 1.2 Using Skill
 - 1.3 Using Statistic
 - 1.4 Comparison Symbols
 - 1.5 Logic Operators

You can route interactions to the most appropriately skilled agent using a skill **expression** or a statistical expression. Skill Expression Builder lets you create both types of expressions, which produce a result of true or false.

Also see: **Variables and Literals**

Using Skill Expression Builder

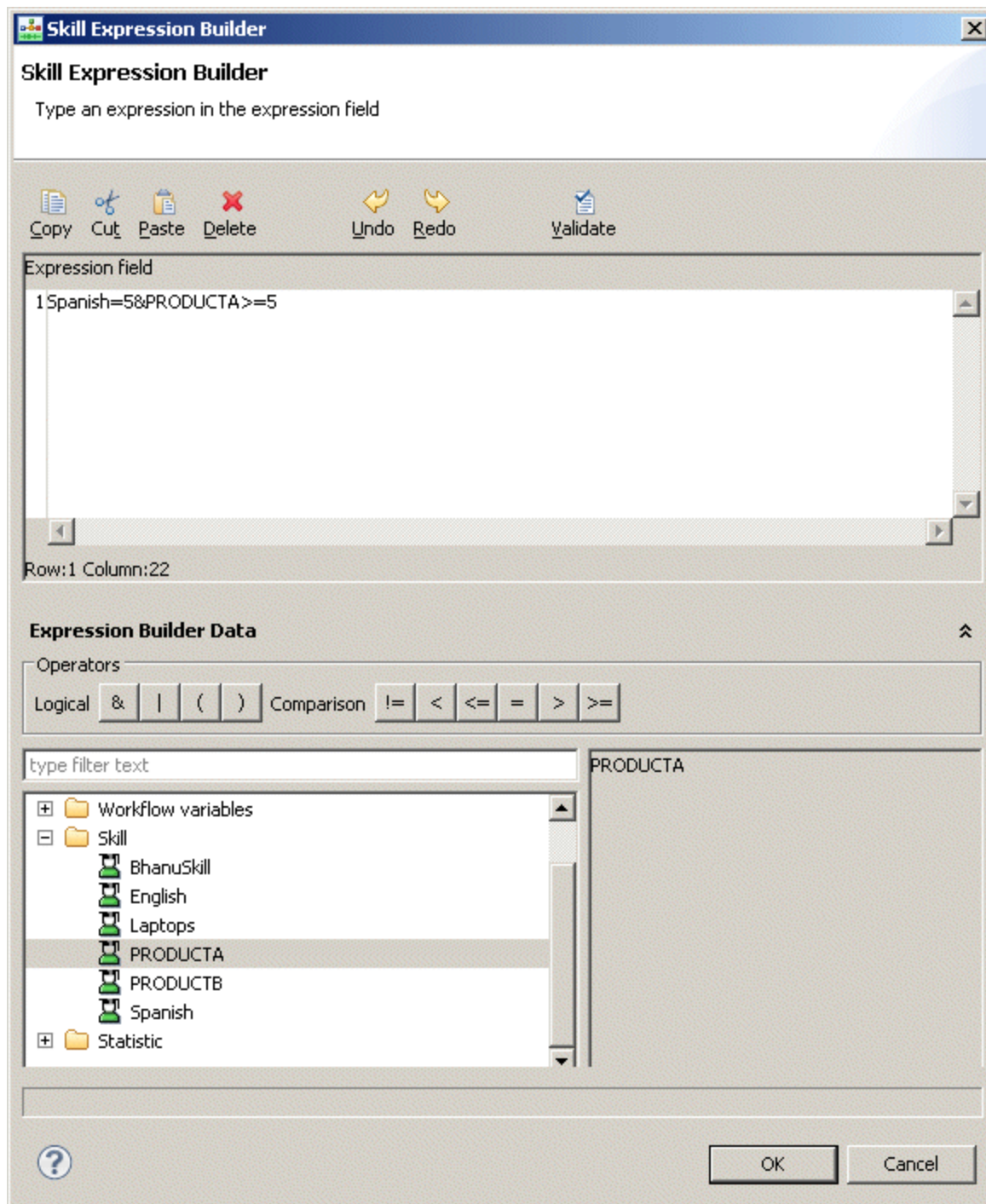
Open Skill Expression Builder from the Targets property in the **Target** block as follows:

1. If you have not already done so, **Server connect to Configuration Server**. Otherwise, when selecting a Target of type Skill, the Skill Expression Builder will not be available.
2. Set the **Validate Skill Expressions** preference.
3. Opposite the Targets property, click under Value to display the  button.
4. Click the  button. The Targets dialog box opens.
5. Click Add in the Targets dialog box.
6. Click under Type to display a down arrow.
7. Click the down arrow and select the Skill target type.
8. Click under the Name field to display the  button.
9. Click the  button to bring up the Skill Expression Builder. Skill and Statistic appear in the lower text area.
 - **Skill**—The name a skill defined in the Configuration Database. Skill names are limited to alphanumeric characters and underscores, cannot begin with a digit, and cannot exceed 126 characters.
 - **Statistic**—The name a statistic/metric defined in the Configuration Database. The statistic name in a skill expression can be any agent statistic used in URS function SData, which returns the current value of the statistic for a given target. For example, you may wish to have URS return the number of interactions waiting, so that if a target is not available, the caller will hear the IVR announce the number of interactions ahead of him.

The statistic must be written in the format: `$(statisticname)`. For example: `$(StatAgentLoggedIn)=1`

Using Skill

1. Expand **Skill**. Skill objects in your Configuration Database are listed for selection.
2. Select a Skill.
3. Create the skill **expression** using a combination of the **Comparison symbols**, **values**, variables, and the & | (, and) logic operators. An example is shown below.



4. Click the button to validate on the toolbar. For example, the expression

Spanish > 5 & ProductA >= 5

was created as follows:

- Expand **Skill**. Double-click **Spanish**. Click the > symbol to insert. Type the number **5** followed by a space. Click the button for the **&** logic operator. Type a space. Double-click **ProductA**. Click the >= symbol. Type the number **5**.
- Click **OK**.

Using Statistic

1. Expand **Statistic**. URS predefined statistics appear for selection.
2. Select a **statistic**.
3. Click a comparison symbol.
4. Create the expression using a combination of the **comparison symbols**, **values**, and the **& | (, and) logic operators**. Example:

`$(RStatCallsInQueue) = 3 & $(RStatCost) = 2 | $(PositionInQueue)) >=6.`

Note: Use of the RStatCost statistic requires that you have cost-based routing implemented at your site. For a description of each statistic, consult the Universal Routing 8.1 Reference Manual.

5. Click the button to validate on the toolbar.
6. Click **OK** when through creating the expression to return to the Targets dialog box.

Comparison Symbols

The table below describes the comparison symbols used to evaluate a skill condition.

Symbol	Interpretation
!=	Differs depending on the Data type: Skill-not equal to the indicated level value. Statistic-not equal to the indicated statistic value.
<	Differs depending on the Data type: Skill-less than the indicated level value. Note: depending on how you use this operator, it may result in including agents that do not have the skill at all (skill name = 0). For example, with English < 8, the queue functional module includes all agents with the English skill less than 8, and also agents with no English skill at all. Statistic-less than the indicated statistic value
<=	Differs depending on the Data type: Skill-less than or equal to the indicated level value. Statistic-less than or equal to the indicated statistic value
=	Differs depending on the Data type: Skill-equal to the indicated level value. Statistic-equal to the

	indicated statistic value.
>	Differs depending on the data type: Skill-greater than the indicated level value. Statistic-greater than the indicated statistic value
>=	Differs depending on the Data type: Skill-greater than or equal to the indicated level value. Statistic-greater than or equal to the indicated statistic value.

This is a value of the same data type as the Data name element. The value must already evaluate to an integer. Float numbers are not supported. There are different limitations depending on the data type:

- **Skill value**—This value represents the level of skill. For example; an agent could have an English skill level greater than 3 (English > 3). An agent can be excluded from a skill by setting that agent's skill level for that skill to zero in the configuration layer (English=0).
- **Statistic**—This value represents the value of the statistic/metric. For example; an agent could be in Ready state longer than 20 seconds ($\$(StatTimeInReadyState) > 20$).

Logic Operators

Use the logic operators to evaluate multiple conditional expressions together. The following logic operators are supported:

- **AND** (&)
- **OR** (|).

The AND and OR logic operators have the same priority. For example:

English >3 & $\$(StatAgentLoggedIn)=1$

Variables and Literals

Starting with 8.1:

- Composer supports variables in skill expressions (they appear in the Skill Expression builder tree).
- You must enclose literal expressions in single quotes.

Background: Previously, Composer automatically added single quotes around the expression entered by the user. Now that variables are supported, Composer must distinguish literal strings and variables. As a result, you must enclose literal strings in single quotes.