Composer Help

Assign Common Block

5/11/2025

Assign Common Block

Contents

Use the Assign common block to assign a computed value/expression or an entered value to a variable.

See the Query Services block Service Data property for an example of using the Assign block and Expression Builder to parse a JSON string and assign the service data to a variable.

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. By default, this option is disabled for backward compatibility. To set this preference, right-click the Project, select **Properties**, **Default Logging** and check **Log Assign block Variable assignments**. Applicable for both Java and .NET Projects.

Starting with 8.1.440.18, Composer Assign blocks are enhanced to generate logging statements as part of code generation. With this enhancement ORS and MCP logs will show the Assign variables and expressions.

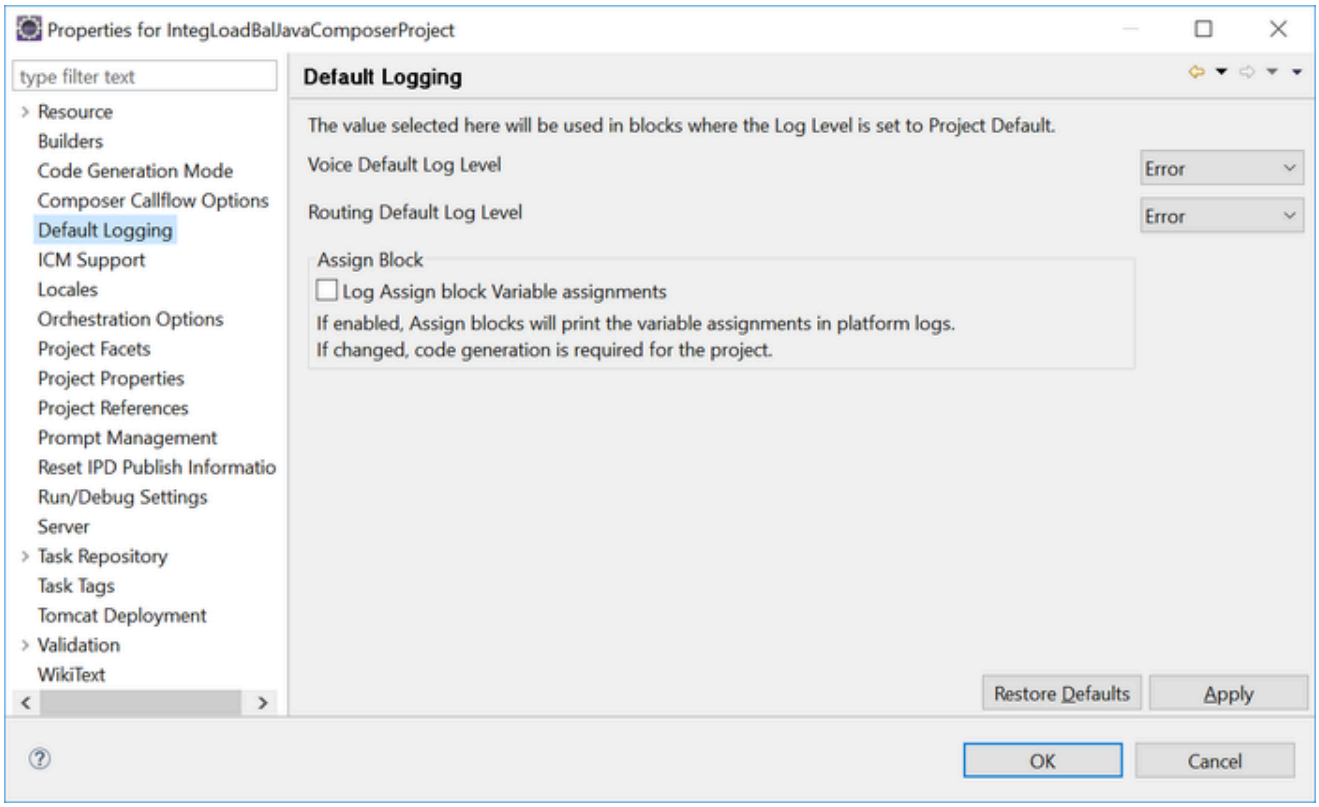
VXML script

```
<assign name="AppState.var0" expr="'This is a callflow diagram file'" />
<log expr="'Assigning variable AppState.var0 with value ' +'This is a callflow diagram file'"/>
```

SCXML script

```
<script>
        var0 = 'Welcome to workflow diagram';
        __Log('Assigning variable var0 with value ' +'Welcome to workflow diagram');
</script>
```

A new Project-level property, Default Logging, is added to control this logging capability. By default, this option is disabled for backward compatibility. Applicable for both Java and .NET Projects.



The Assign block has the following properties:

Name Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.

Block Notes Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.

Assign Data Property

This property assigns a value (expression) to a variable. You select the variable and then enter an expression, either a literal or one created in Expression Builder.

To select a variable and assign a value:

1. Click the **Assign Data** row in the block's property table.
2. Click the  button to open the Assign Data to Variables dialog box.
3. Click in the **Variable** field to display a down arrow.
4. Click the down arrow and select a variable whose value will be evaluated to determine the branching condition. Default application variables are described in the Entry block for voice applications and the Entry block for routing applications. You can also use a custom variable.
5. Click under Expression to display the  button.
6. Click the  button to open Expression Builder. For examples of how to use Expression Builder, see the Expression Builder topic.
7. Select an operator for the branching condition.Your variable's value will be equal to (==), less than (<), greater than (>). less than or equal to (<=), greater than or equal to (>=) or not equal to (!=) to value you enter in the Expression field.
8. In the Expression field, create a value to compare to the variable's value. Enclose the value in single quotes (' ').
9. Click the  button to validate the expression. Syntax messages appear under the Expression Builder title.
10. Click **OK** to close Expression Builder and return to the Assign Data to Variables dialog box.
11. You can make multiple variable/value assignments. Click the **Add** button if you wish to add more assignments and repeat the steps above.

Editing Expressions

To edit an expression:

1. Click its row under Expression in the Assign Data to Variables dialog box. This causes the  button to appear.
2. Click the  button to re-open Expression Builder where you can edit the expression.

Exceptions Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.

- For callflows, invalid ECMAScript expressions may raise the following exception event: `error.semantic`.
- For workflows, invalid ECMAScript expressions may raise the following exception events: `error.script.SyntaxError`, and `error.script.ReferenceError`.

You can use custom events to define the ECMAScript exception event handling.

Condition Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.

Logging Details Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.

Log Level Property

Find this property's details under CommonProperties for Callflow Blocks or Common Properties for Workflow Blocks.

Enable Status Property

Find this property's details under Common Properties for Callflow Blocks or Common Properties for Workflow Blocks.