



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

[Set Ideal Agent Block](#)

# Set Ideal Agent Block

## Contents

- **1 Set Ideal Agent Block**
  - 1.1 Name Property
  - 1.2 Block Notes Property
  - 1.3 Exceptions Property
  - 1.4 Condition Property
  - 1.5 Logging Details Property
  - 1.6 Log Level Property
  - 1.7 Interaction ID Property
  - 1.8 ORS Extensions Property
  - 1.9 Enable Status Property
  - 1.10 Target Property

Starting with Composer release 8.1.410.14, workflow diagrams add a Set Ideal Agent block to the Routing palette, which allows you to select the most ideal agent to handle an interaction when more than one agent is available. You can also use this functionality to select the most ideal interaction when there is more than one interaction competing for the same agent. The Composer Set Ideal Agent block invokes the Universal Routing Set Ideal Agent functionality described in [Using Agent Skills for Ideal Agent Selection](#).

This block has two specific Set Ideal Agent properties: **Interaction ID** (default: system.InteractionID) and **Target** (provides support for Skill Expression Builder).

### Important

In order to use this block, you must have installed Universal Routing Server release 8.1.400.22 or later.

## Name Property

Find this property's details under [Common Properties](#).

## Block Notes Property

Find this property's details under [Common Properties](#).

## Exceptions Property

Find this property's details under [Common Properties](#). The `error.session.fetch` exception is supported.

## Condition Property

Find this property's details under [Common Properties](#).

## Logging Details Property

Find this property's details under [Common Properties](#).

### Log Level Property

Find this property's details under [Common Properties](#).

### Interaction ID Property

Set to a meaningful value or keep the default value, which is the system variable `system.InteractionID`. Can be used for interaction-less processing for scenarios where the `InteractionId` variable is not automatically initialized, but instead must wait for an event. An example would be an SCXML application triggered by a Web Service that does not add an interaction.

### ORS Extensions Property



Starting with release 8.1.4, Composer blocks used to build routing applications (with the exception of the Disconnect and EndParallel blocks) add a new [ORS Extensions](#) property.

### Enable Status Property

Find this property's details under [Common Properties](#).

### Target Property

Use this property to specify a skill expression.

1. If you have not already done so, [connect to Configuration Server](#).
2. Opposite the **Target** property, click under **Value** to display the  button.
3. Click the  button. Expression Builder opens with **Target** selected and available Skills listed below.
4. Construct a skill expression to select the ideal agent as described in [Using Agent Skills for Ideal Agent Selection](#).

### [+] Set Ideal Agent Code Sample

```
<state id="SetIdealAgent">
  <onentry>
    <log expr="_sessionid + ': Inside Set Ideal Agent Block: SetIdealAgent'" />
    <script>var skillexpr = ["Billing >10"];</script>
    <session:fetch requestid="App_SetIdealAgent['requestid']" srcexpr="'urs/call/@' +
system.InteractionID + '/func/SetIdealAgent'" timeout="30" method="'urs'">
      <param name="params" expr="uneval(skillexpr)" />
    </session:fetch>
  </onentry>
</state>
```

## Set Ideal Agent Block

---

```
    </session:fetch>
  </onentry>
  <transition event="session.fetch.done"
cond="_event.data.requestid==App_SetIdealAgent['requestid']" target="$_MY_PREFIX_$.Exit1">
    <log expr="'Composer Application:default Block: SetIdealAgent'" />
    <log expr="'Session FETCH DONE'" />
    <script>storeEvent("SetIdealAgent", _event);</script>
  </transition>
</state>
```