



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

User Data Block

# User Data Block

## Contents

- **1 User Data Block**
  - 1.1 Important Notes
  - 1.2 Name Property
  - 1.3 Block Notes Property
  - 1.4 Assign Data Property
  - 1.5 Condition Property
  - 1.6 Logging Details Property
  - 1.7 Exceptions Property
  - 1.8 Interaction ID Property
  - 1.9 Log Level Property
  - 1.10 Enable Status Property
  - 1.11 ORS Extensions Property
  - 1.12 Internal Key Prefix
  - 1.13 Wait For Event Property
  - 1.14 Timeout Property

You can use in a routing application to update an interaction's **User Data** and for attaching Business Attributes, Categories and Skills. Corresponds to function `_genesys.ixn.setUserData` under Functions in the *Orchestration Developer's Guide* and available in **Expression Builder**. This block generates ECMAScript inside an SCXML state and does not rely on **External Service Protocol** via `<session:fetch>`. For manually attaching categories to an interaction, the User Data block can be used and then a **Branching block** can be (optionally) used to segment interactions to different logical branches based on the different categories. Important! **See Mandatory User Data For UCS Blocks.**

## Important Notes

- Do not assign the value of a variable named data to a key-value pair. This will not work since the generated code also declares a variable named data.
- When the Wait for Event property is set to true and User Data blocks are used in both the parallel legs, use **Internal Key Prefix** to reliably verify the user data attachment.

The User Data block has the following properties:


## Name Property

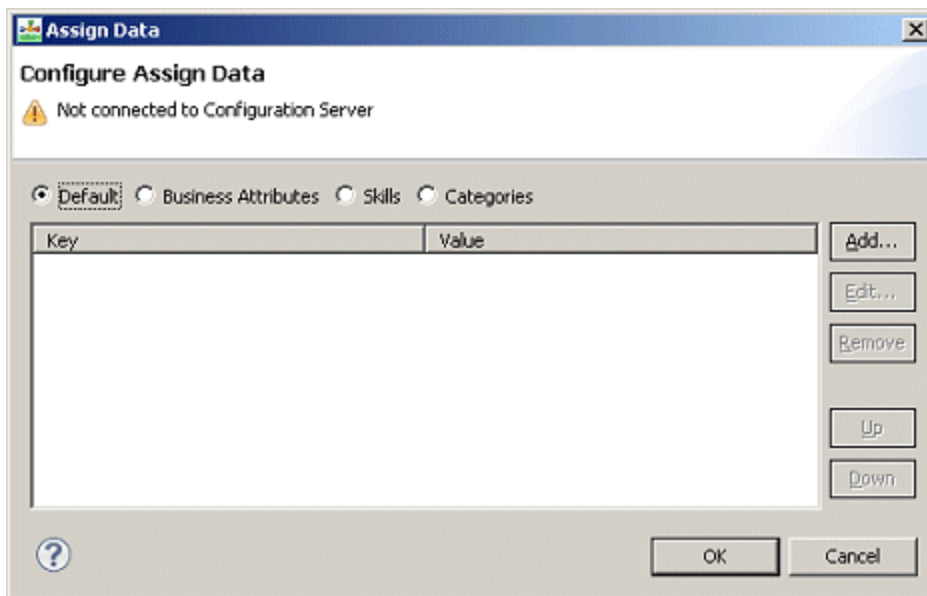
Find this property's details under **Common Properties**.

## Block Notes Property

Find this property's details under **Common Properties**.

## Assign Data Property

This property specifies one or more key-value pairs to add to the interaction's User Data. To specify key-value pairs, click in the Value column to display the  button and click it to bring up the dialog. Data from various sources can be attached as well as free form key value pairs can be specified (Default). However, **note that only one category can be used in a block. Switching categories will erase previously specified values.**



UserData1.gif

1. **Default** This option can be used to specify both the key name and the key value as literals or variables.

**Note:** In case the key or value contains special characters that may require encoding e.g. '<', '-', or quotes, define a variable and set its value to this literal and use the variable as the value.

2. **Business Attributes** This picks up specific business attributes and if connected to Configuration Server, shows a list of values configured for these attributes.
3. **Skills** One or more skills can be specified. If connected to Configuration Server, a list of skills is shown.
4. **Categories** This option requires an active connection to **Configuration Server** and supports attaching categories defined in Configuration Server as well as **Relevancy** for the category. **Relevancy** is a number from 1 to 100 that reflects the minimum relevance percentage that each classification category must have in order for **Classification Server** to consider an interaction as belonging to that category.

Click the Add button to add a key-value pair of the selected type. When done, click Ok to dismiss the dialog and Save the diagram.

## Condition Property

Find this property's details under **Common Properties**.

## Logging Details Property

Find this property's details under **Common Properties**.

---

## Exceptions 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 `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. **Background:** Previous to 8.1.1, Composer did not expose an `InteractionId` property. Instead, when ORS started processing an interaction, a generated SCXML application automatically initialized the system variable, `InteractionId`. This variable was then used internally by Routing and certain eServices blocks when interacting with ORS. With the introduction of support for Interaction-less processing, you can now define a specific event to initialize `InteractionId`, or not define an event at all. For scenarios with an interaction (IPD Diagram/Wait For Event=`interaction.present` for example), you may keep the default value for the Interaction ID property. The default value is the system variable `InteractionId`, which is initialized automatically in this case. For other scenarios (any scenario where the system variable `InteractionId` is not set), you may choose to:

1. Not use blocks that require an Interaction ID
2. And/or set the Interaction ID property to a meaningful value
3. And/or assign a meaningful value to the `InteractionId` system variable

## Log Level Property

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

## Enable Status Property

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

## ORS Extensions Property

Starting with 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.

## Internal Key Prefix

Starting with 8.1.420.14, Composer attaches an internal key along with the configured user data. When the Wait for Event property is set to true and User Data blocks are used in both parallel legs, use Internal Key Prefix (the **Show Advanced Properties** button) to reliably verify the user data attachment. The value of the internal key is the time stamp of the application change. This key is used internally to verify whether the `interaction.udata.changed` event has been received. If parallel User Data blocks are used in a workflow, the internal keys might mismatch, which leads to a timeout of User Data blocks. You can configure the internal key prefix either directly through this property or through variables. The configured value will be attached as a prefix to the existing Composer-generated internal key; for example:

```
Composer_<Internal Key Prefix>_internal_key = <timestamp>
```

### Important

Use the Internal Key Prefix only when setting user data from two parallel legs and waiting for confirmation in both legs.

## Wait For Event Property

This property allows you to choose whether to wait for the user data changed event before transitioning to the next block.

1. Click **Wait for Event** under **Property**.
2. Under **Value**, select one of the following:
  - **True**
  - **False**

## Timeout Property

Select the variable that contains the timeout value for the user data change event. This property supports the following:

- Literal integer value. For example: `Timeout=4 & Unit=second <send event="'WaitEvent1.wait.timeout'" delay="'4s'"/>`.
- Variable with integer value. For example: `Timeout=Variable(4) & Unit=second <send event="'WaitEvent1.wait.timeout'" delay="'4s'"/>`.
- Variable with string value. For example: `Timeout=Variable('6') & Unit=second <send event="'WaitEvent1.wait.timeout'" delay="'6s'"/>`.
- Variable with string value including unit. For example: `Timeout=Variable('6ms') & Unit=second <send event="'WaitEvent1.wait.timeout'" delay="'6ms'"/>`. In this case, the unit specified in the

variable is used instead of the static property unit.