



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

Query Services Block

Query Services Block

Contents

- [1 Query Services Block](#)
 - [1.1 Use Case](#)
 - [1.2 Name Property](#)
 - [1.3 Block Notes Property](#)
 - [1.4 Service Elements Property](#)
 - [1.5 Extensions Property](#)
 - [1.6 Exceptions Property](#)
 - [1.7 Condition Property](#)
 - [1.8 Logging Details Property](#)
 - [1.9 Log Level Property](#)
 - [1.10 Enable Status Property](#)
 - [1.11 Service Data Property](#)
 - [1.12 Variables Mapping Property](#)
 - [1.13 Identifier Property](#)
 - [1.14 Service Status Property](#)
 - [1.15 Service Type Property](#)
 - [1.16 Service Completed After Property](#)
 - [1.17 Service Completed Before Property](#)
 - [1.18 Service Started After Property](#)
 - [1.19 Service Started Before Property](#)

Use this block to query the [Universal Contact Server Database](#) for a list of [services](#) associated with a particular Customer ID or, in case of unassociated services, the Contact Key. Composer stores the result in an application variable. You can query for:

- Active services
- Completed services
- Both active and completed services

Use Case

[Service](#)/state history is primarily meant to support service personalization and resumption. For example, a given application is using a state to record the fact that the system is waiting for the customer to fax in a signed authorization to complete a transaction. When the customer calls into the IVR to verify some recent activities, the IVR application queries the service state history and is informed that a service is waiting on a fax to arrive. The Query Services block has the following properties: Note! The behavior of some properties can vary depending on whether you are in [offline](#) or [online mode](#).

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](#).

Service Elements Property

Use this property to indicate whether information on completed service elements/tasks should be included in the returned results.

1. Click under **Value** to display the  button.
2. Click the  button to open a dialog box.
3. Check one or more of the following:
 - **Active Services**
 - **Completed Services**

- **Active Tasks**
- **Completed Tasks**

Extensions Property

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

Exceptions Property

Find this property's details under [Common Properties for Callflow Blocks](#) or [Common Properties for Workflow Blocks](#). You can also define [custom events](#).

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](#).

Enable Status Property

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

Service Data Property

Click the down arrow and select a variable to contain the output data for matching services. The output of the Query Service block will be in JSON format. You will need to use the [Assign](#) block and EMCAScript in [Expression Builder](#) to parse the JSON. Below is an example of a small ECMAScript that will take the output of the Query Service block, which has been put into an application variable



"QueryServices," and retrieves the service_id for the customer, if it any active services exist. It puts that service_id value into a workflow application variable called "serviceid". if (QueryServices.length > 0) { serviceid = QueryServices[0].service_id; } else { serviceid = 'new service'; } **Note:** You can also do the same things with Composer's **Variables Mapping** feature. It is easier (and it saves an ECMAScript block) to simply use the Variables Mapping property to map "service_id" to the application variable "serviceid".

Variables Mapping Property

Use this property to map the JSON data returned by this block to variables. See the **Variables Mapping** topic for details.



Identifier Property

Use this property to identify the customer. Choose the Customer ID (for associated services) or the Contact Key (for unassociated services).

1. Click under **Value** to display the  button.
2. Click the  button to open the Identifier dialog box.
3. Select one of the following buttons:
 - **Customer Identifier** (for associated services).
 - **Contact Key** (for anonymous services)
1. Click the down arrow opposite **Type** and select the source:
 - **Literal**. Then enter the attribute, such as CustomerID.
 - **Variable**. Then select the variable that contains the Contact Key or Customer ID.

Service Status Property

Use this property to choose the Composer method to call.



1. Click under **Value** to display the  button.
2. Click the  button to open the Service Status Selection dialog box.
3. Opposite Type, select one of the following:
 - **Variable**. Then click the **Value** down arrow and select a variable with a value of "Completed", "Active", or "All".
 - **Literal**. Then select one of the following: **Completed**, *Active*, **All**

Service Type Property

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

Service Completed After Property



Use this property to filter for services completed on or after the given date/time.

1. Click under **Value** to display the  button.
2. Click the  button to open the Service Completed After dialog box.
3. Click the down arrow opposite **Type** and select one of the following:
 - **Literal**. Click arrows to select the Date and Time or manually enter.
 - **Variable**. Select the variable whose value must be formatted with the ISO 8601 UTC pattern YYYY-MM-DDTHH:mm:ss.SSSZ (for example: 2010-03-15T11:33:48.000Z).

Also see the [Time Zone Preferences](#) topic.

Service Completed Before Property


Use this property to filter for services completed prior to the given date/time. If using a variable for this property, the variable value must follow the ISO 8601 UTC pattern: [YYYY]-[MM]-[DD]T[HH]:[mm]:[ss].[SSS]Z.


1. Click under **Value** to display the  button.
2. Click the  button to open the Service Completed Before dialog box.
3. Click the down arrow opposite **Type** and select one of the following:
 - **Literal**. Click arrows to select the Date and Time or manually enter.
 - **Variable**. Select the variable whose value must be formatted with the ISO 8601 UTC pattern YYYY-MM-DDTHH:mm:ss.SSSZ (for example: 2010-03-15T11:33:48.000Z).

Also see the [Time Zone Preferences](#) topic.

Service Started After Property

Use this property to filter for services started on or after the given date/time.



1. Click under **Value** to display the  button.

2. Click the  button to open the Service Started After dialog box.
3. Click the down arrow opposite **Type** and select one of the following:
 - **Literal.** Click arrows to select the Date and Time or manually enter.
 - **Variable.** Select the variable whose value must be formatted with the ISO 8601 UTC pattern YYYY-MM-DDTHH:mm:ss.SSSZ (for example: 2010-03-15T11:33:48.000Z).

Also see the [Time Zone Preferences](#) topic.

Service Started Before Property

Use this property to filter for services started prior to the given date/time.

1. Click under Value to display the  button.
2. Click the  button to open the Service Started Before dialog box.
3. Click the down arrow opposite **Type** and select one of the following:
 - **Literal.** Click arrows to select the Date and Time or manually enter.
 - **Variable.** Select the variable whose value must be formatted with the ISO 8601 UTC pattern YYYY-MM-DDTHH:mm:ss.SSSZ (for example: 2010-03-15T11:33:48.000Z).