Service Manager Connection Package - Use Case

TABLE OF CONTENTS

Contents

From: Service Manager versions **Oxygen 2.0** and later.

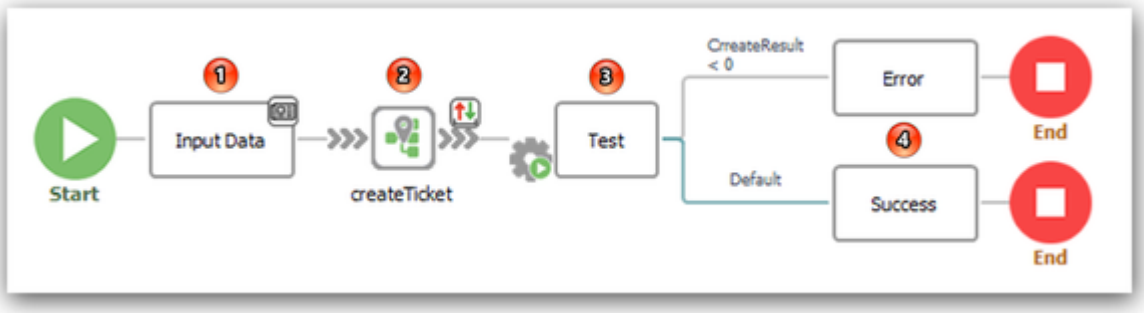
To help you set up this use case, you can download the relevant documentation and sample project. See List of files to download.

This use case enables you to create a ticket in Service Manager from a Self Help procedure. It uses the *createTicket* endpoint in the Service Manager connection package.

OVERVIEW

The implementation of the use case is performed in several phases:

- Phase 1: Collect the information required for creating the ticket **(1)**
  - Retrieve the ticket catalog code.
  - Retrieve the email address of the requestor/recipient.
  - Retrieve the description of the ticket using a form.
  - Associate the form with the first data collection step.
- Phase 2: Call the *createTicket* endpoint **(2)**
  - Create a sub-procedure associated with the *createTicket* endpoint.
  - Configure the input and output parameters of the endpoint.
- Phase 3: Define the conditions for creating the ticket **(3)**
  - Create an *Action with Switch* step to manage two possibilities, *Ticket creation successful and Ticket creation failed*.
  - Create a failure condition: the status of the request is negative.
- Phase 4: Process the result of the ticket creation test **(4)**
  - Create a *Page* step to process the *Ticket creation failed* result (failure condition fulfilled).
  - Create a *Page* step to process the *Ticket creation successful* result (failure condition not fulfilled).



PREREQUISITES

1. Install and configure the Service Manager connection package.

   See the procedure.

2. Create a new procedure in your Self Help project.

   See How to create a Self Help procedure.

PHASE 1: COLLECT THE INFORMATION REQUIRED FOR CREATING THE TICKET

See the description of the *createTicket* endpoint.

Information to be collected:

- Ticket catalog code. This can be obtained in three ways:
  - Using the *Category* metadata of the procedure
  - Using the ticket category code corresponding to *Category to qualify*
  - Using a form completed by the user
- Email address of the requestor/recipient, i.e. email address of the logged-in user
- Description of the ticket, using a form completed by the user

Step 1: Create a step called *Input data*.

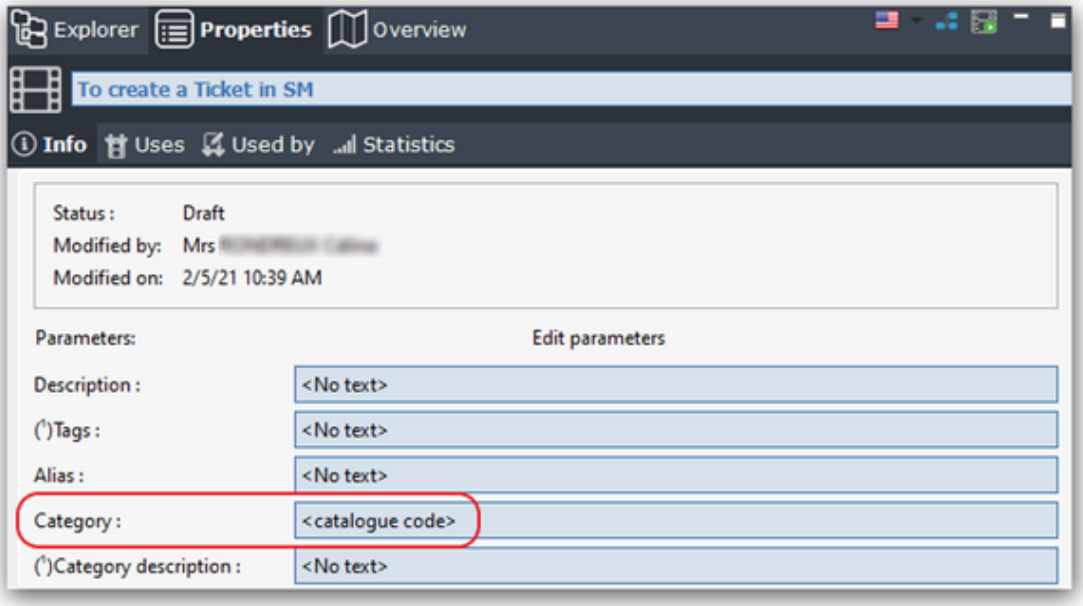
1. Insert a *Page* step in your Self Help procedure.
2. Name it *Input data*.

Step 2: Retrieve the ticket catalog code.

Note: Use one of the three methods below to retrieve the catalog code.

1. You want to use the *Category* metadata of the procedure.
   - Display the metadata of your new procedure in the *Properties* view.

- Enter the catalog code of the new ticket in the *Category* metadata.



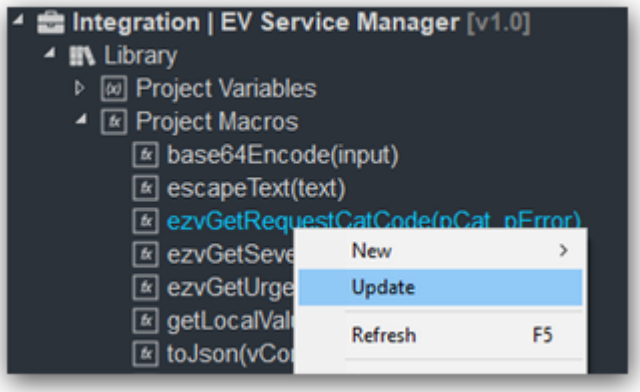
2. You want to use the default ticket catalog code.

Note:

- The default catalog code is automatically loaded when you create a ticket without a catalog code.
- The connection package is shipped with default catalog code *58* which corresponds to the *Get Help* category.
- You can modify the default code using the *ezvGetRequestCatCode* macro available in the project library.

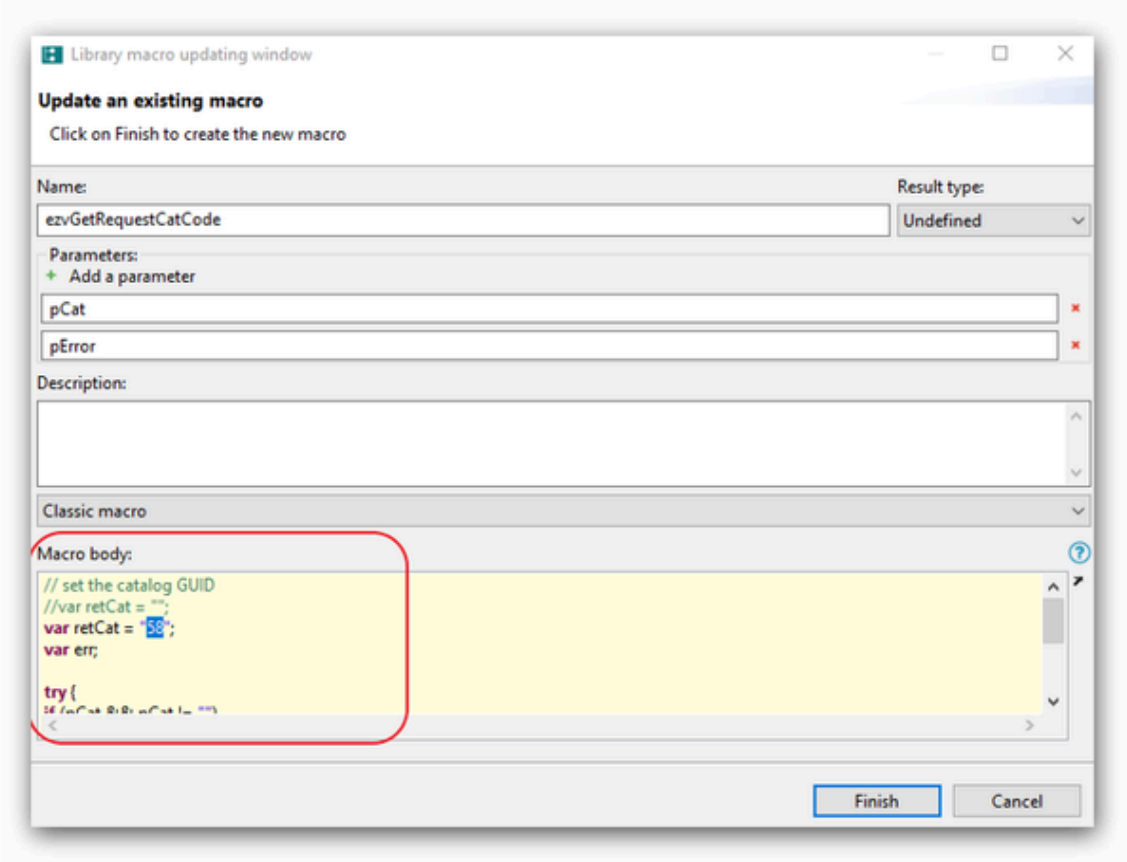
Modify the default catalog code

- Expand the tree structure of the connection package.
- Go to the **Project Macros** section in the *Library* folder.
- Right-click the *ezvGetRequestCatCode* macro and select *Update* from the contextual menu.



- Enter the new default catalog code.

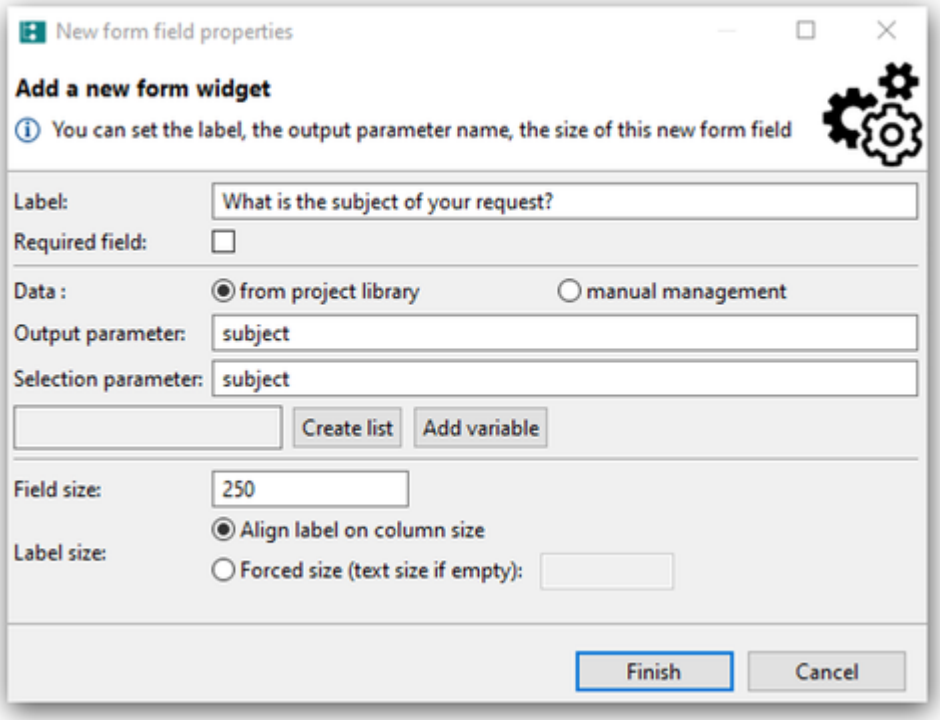
- Click Finish.



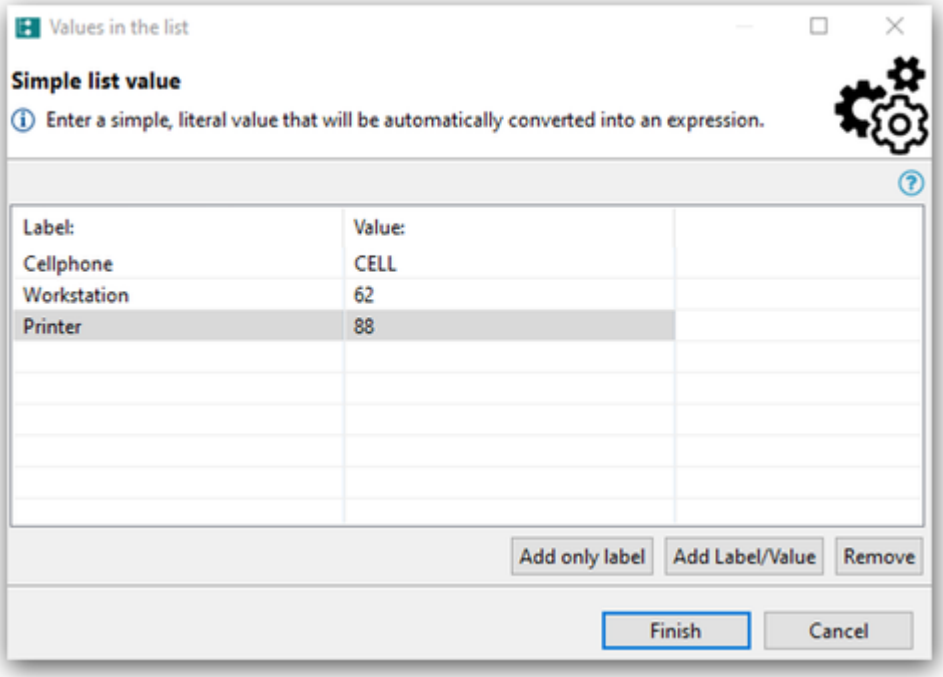
3. You want to ask the user to enter the catalog code in a form.

- Create a new form in your Self Help project.
- Name it *fTicketCreation*.

- Add a *Simple choice list of values* field.



- Click Create list.



- Specify the catalog code values to be proposed to users.
  - Click Add Label/Value.
  - Enter the description of the catalog code in the **Label** field.
  - Enter the value of the catalog code in the **Value** field.
- Click Finish.

Step 3: Retrieve the email address of the requestor/recipient.

The email address is automatically retrieved using the email address of the logged-in user.

Caution: Ensure that the email address specified in Self Help is identical to the one specified in Service Manager.

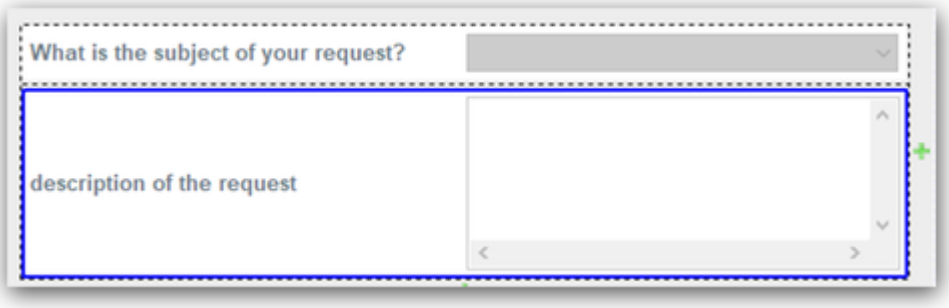
Step 4: Retrieve the description of the ticket.

1. Create a form if you did not create one for obtaining the catalog code. Name it *fTicketCreation*.

or

Use the *fTicketCreation* form you created earlier for obtaining the catalog code.

2. Add a text field with multiple lines where users can enter a description.



Step 5: Associate the form with the *Input data step*.

1. Click and drag the *fTicketCreation* form from the **Explorer** pane to the *Input data* step.

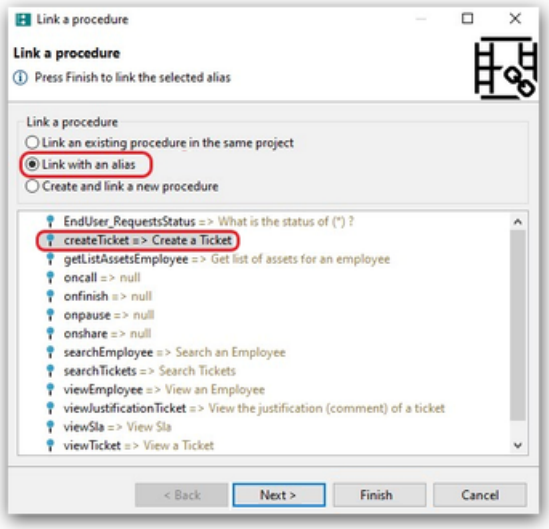
PHASE 2: CALL THE *CREATETICKET* ENDPOINT

Step 1: Create a step for calling the endpoint.

1. Insert a *Sub-procedure* step in the Self Help procedure.

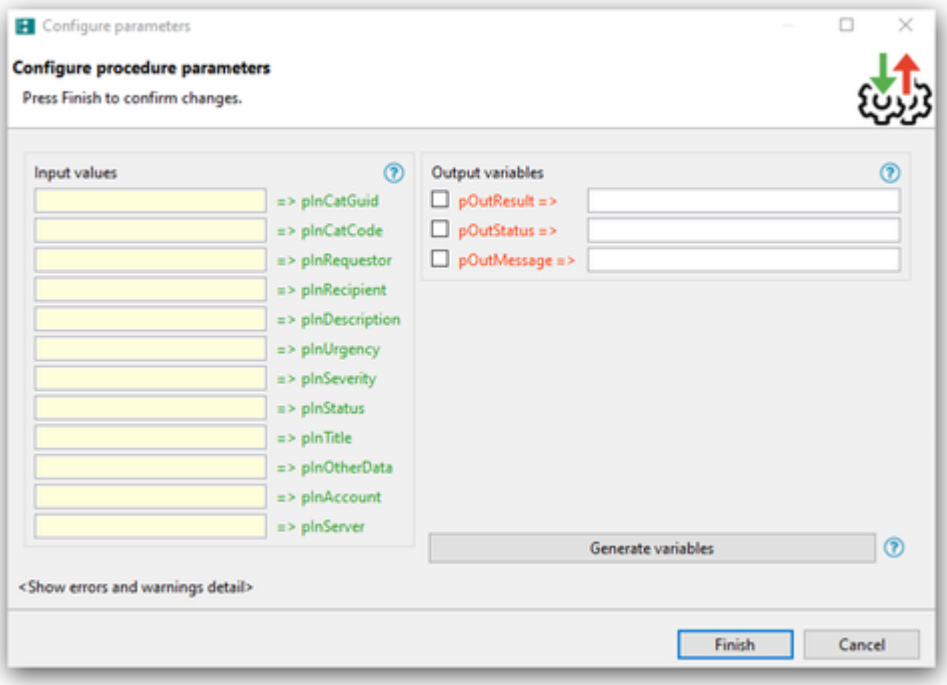
2. Double-click the step.
The properties window of the procedure will appear.

3. Select the *Link to procedure alias* option.
The list of aliases available is displayed.



4. Select the alias of the *createTicket* endpoint.

5. Click Next.
The window for configuring input and output parameters will appear.



Step 2: Enter the values of the input parameters of the endpoint.

Note:

- Specify the values in the **Input values** column. ==> Fields are displayed in **green**.
- Parameters without specified variables are ignored.
- You can associate raw values with input variables.
  - **Caution**: Enter the value, surrounded by double quotes.
  - The value will appear in **blue**.

example "aarensky@evtry.com" => pInRequestor

1. Specify the *pInCatCode* variable for storing the catalog code, depending on the method you are using to retrieve the catalog code.

- If you specified the catalog code using the *Category* metadata of the procedure, leave this variable blank.

  => pInCatCode

- If you used the default catalog code, leave this variable blank.
  Caution: Ensure that the *Category* metadata of the procedure is not specified.

  => pInCatCode

- If you created a form asking users to enter the catalog code, associate the corresponding form variable.

  fTicketCreation.subject => pInCatCode

2. Specify the *pInRequestor* and *pInRecepient* variables for storing the email address of the requestor/recipient.

- Associate the email address of the user logged in to Self Help. This is found in the predefined variable called *executionContext.user.email*.

executionContext.user.email => pInRequestor

executionContext.user.email => pInRecepient

3. Specify the *pInDescription* variable for storing the ticket description.

- Associate the corresponding form variable.

example fTicketCreation.description => pInDescription

Step 3: Enter the values of the output parameters of the endpoint.

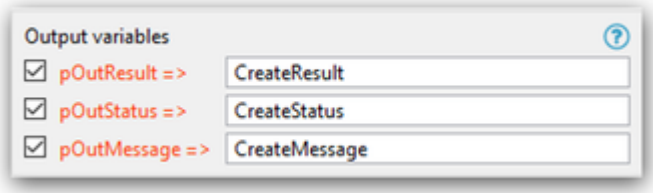
Notes:

- Specifify the values in the **Output variables** column. ==> Fields are displayed in **red**.
- Variables are specific to the current procedure.

1. Select the output parameter.

See the description of output parameters.

2. Enter the name of the associated output variable.



Step 4: Save the step that calls the endpoint.

1. Click Finish.

The step will automatically be renamed *createTicket*.

PHASE 3: DEFINE THE CONDITIONS OF THE TICKET CREATION

Step 1: Create a step for adding the creation conditions.

1. Right-click the *createTicket* step and select *Insert a Step > Insert an Action with Switch* from the contextual menu.

2. Name the step, *Test*.
Two branches will be created.

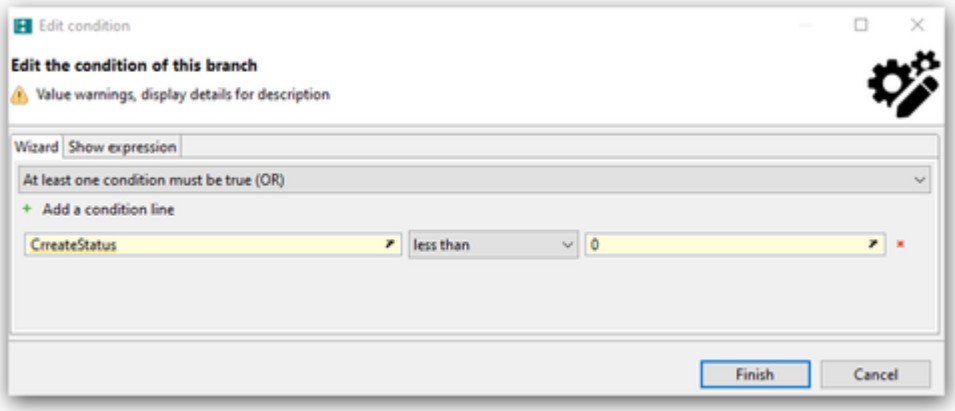
- The top branch *<Condition>* manages the response to be provided in the event of failure of the ticket creation.
- The bottom branch *Default* manages the response to be provided in the event of success of the ticket creation.

Step 2: Define the condition that represents the failure of the ticket creation.

1. Double-click the *<Condition>* field in the top branch.
The window for defining the condition will appear.

1. Enter the condition that indicates a creation failure with a negative status of the query.

- Select the *CreateStatus* output variable that stores the status of the query.
- Select the *less than* operator et enter the value *0*.
  See the description of return codes.



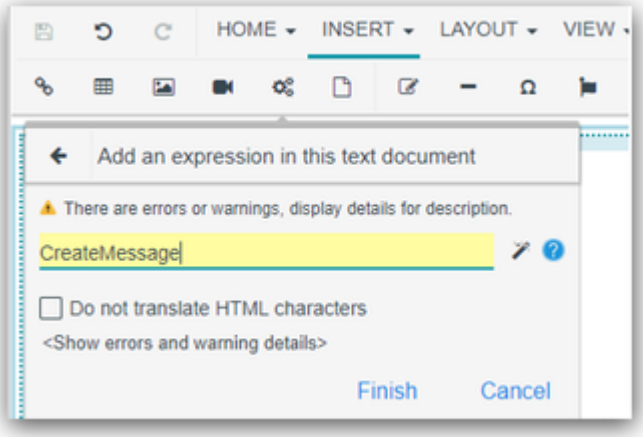
- Click Finish.
  The condition will be refreshed.

PHASE 4: PROCESS THE TEST RESULT OF THE TICKET CREATION

Step 1: Process the result in the event of an error.

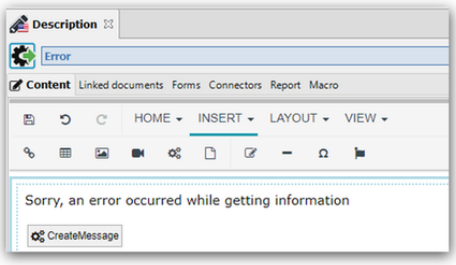
1. Insert a *Page* step after the top branch that is called in the event of an error.
2. Name the step, *Error*.
3. Define the content of the step.

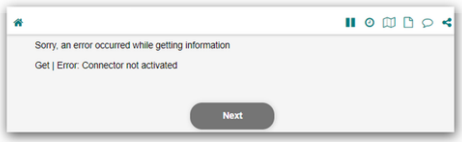
best PRACTICE Click *Insert an expression* and enter the variable name to retrieve the value of variables.



- Enter a message informing users that an error occurred.
- Display the error message using the *CreateMessage* output variable.
  Caution: The name of the variable is case-sensitive.

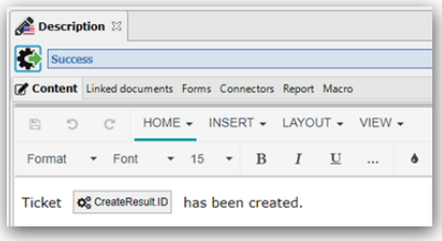
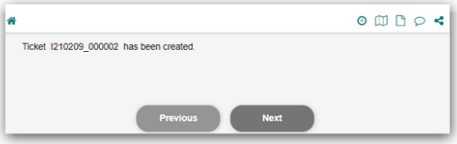
Self Help	Example





Step 2: Process the result in the event of success.

1. Insert a *Page* step after the bottom branch that is called in the event of success.
2. Name the step, *Success*.
3. Define the content of the step.
   - Enter a message informing users that the procedure was correctly executed.
   - Display the new ticket number using the *CreateResult.ID* output variable.
     Caution: The name of the variable is case-sensitive.

Self Help	Example
	

LIST OF FILES TO DOWNLOAD

- PDF file of the use case
- Self Help project archive including the use case
- Service Manager connection package. See the procedure to install the files.