Lab 3 - IDoc Outbound Endpoint

Ver. 1.0.0

Overview

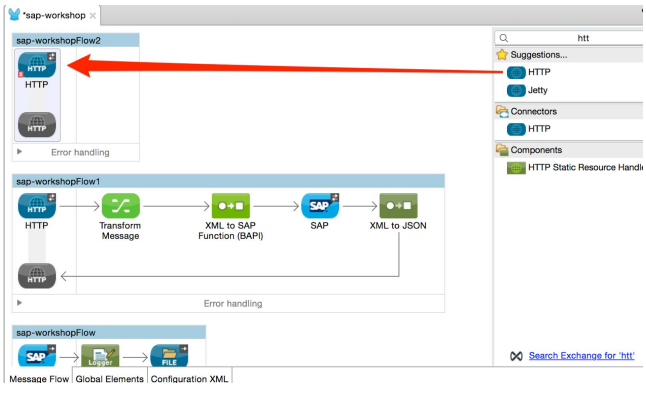
This next lab will focus on using the SAP Connector as an IDoc Outbound Endpoint. The flow that you create will send an IDoc to SAP to create a customer record.

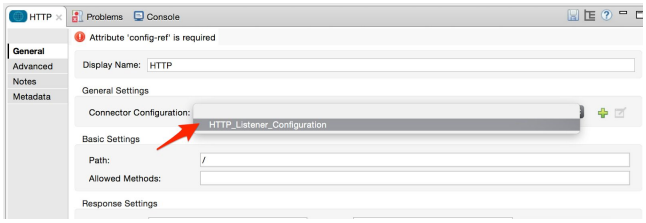
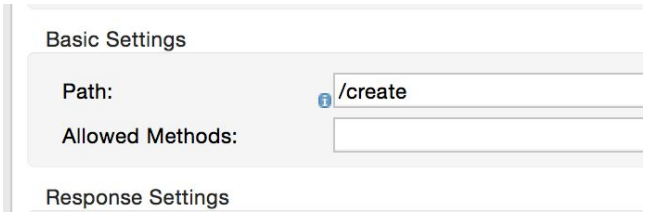
Reminder: This lab requires the use of your own SAP instance. If you don't have one, you can leverage a hosted SAP service. For the purposes of this lab, we utilized Sandbox SAP.

Overview
Steps
1. Setup HTTP Connector
2. Configure SAP Connector
3. Use DataWeave to Create an IDoc
4. Transform XML to SAP IDoc
5. Test Project
Summary

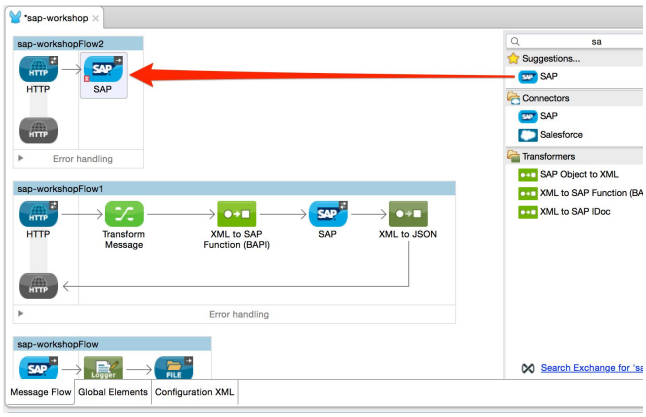
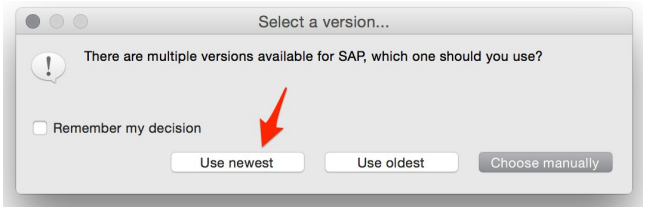
Steps

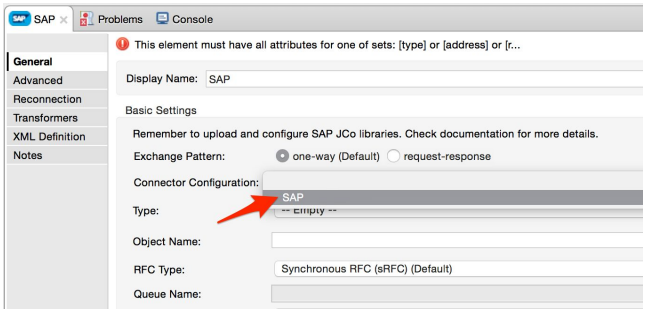
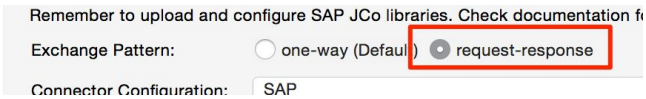
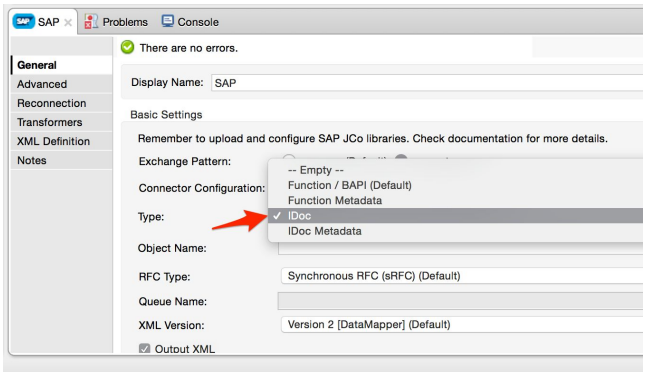
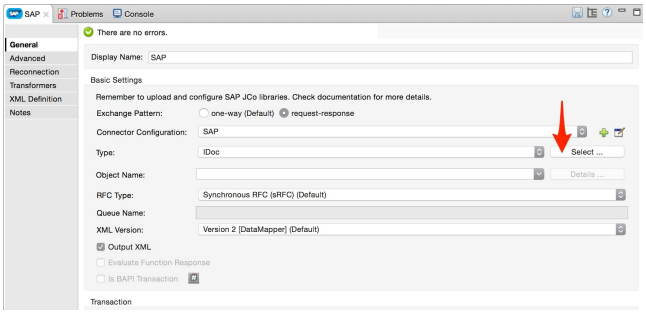
1. Setup HTTP Connector

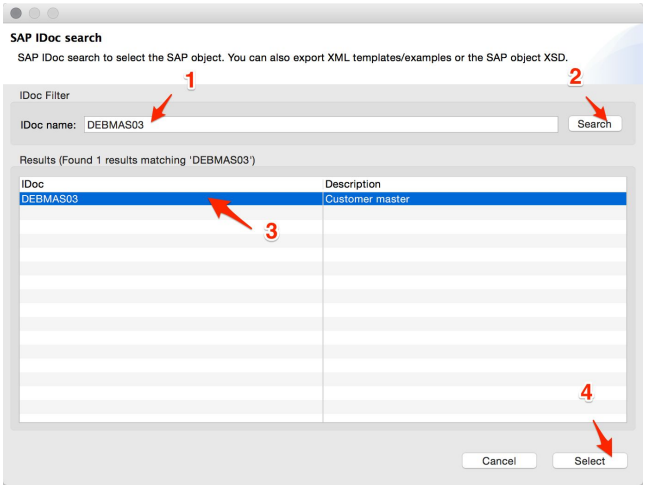
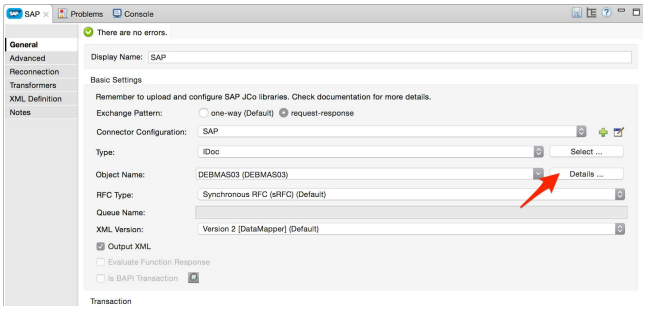
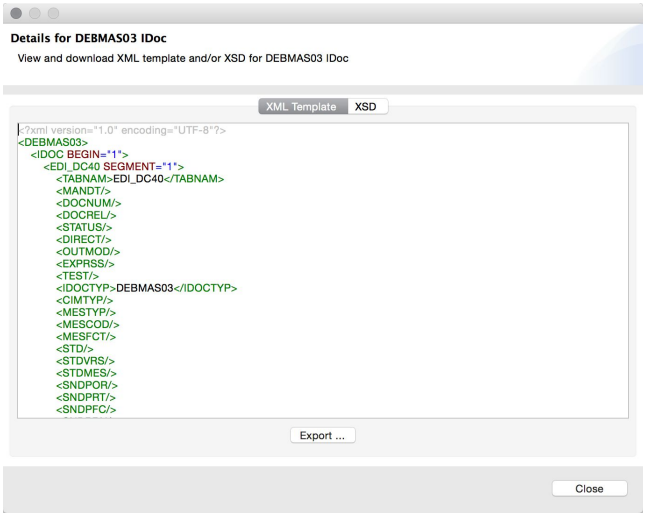
1.1	In the toolkit, search for the HTTP Connector by typing in 'http' in the Search bar. Drag and drop the **HTTP Connector** onto the canvas.	

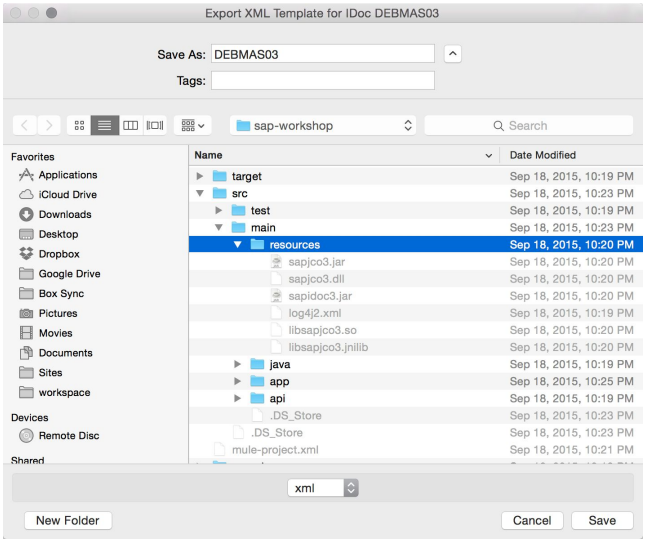
1.2	In the Mule Properties tab for the **HTTP Connector**, select the configuration that was setup in Lab 2. The default should be **HTTP_Listener_Configuration**	
1.3	Under the **Basic Settings** section, set the **Path** field to the following /create When you run the application later, you will use the following URL: http://localhost:8081/create	

2. Configure SAP Connector

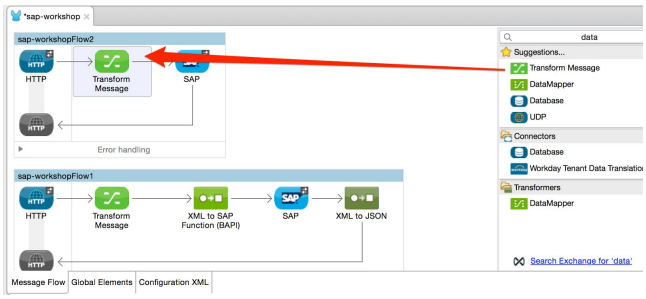
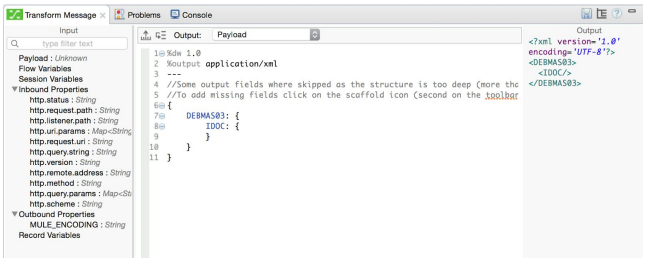
2.1	Next, let's add the SAP Connector. In the toolkit, search for the SAP Connector by typing in 'SAP' in the Search bar. Drag and drop the **SAP Connector** onto the canvas. This will set the **SAP Connector** as an outbound endpoint.	
2.2	If the **Select a version...** window pops up, select **Use newest**	

2.3	In the **Connector Configuration** field, select the configuration that we created in Lab 1. It should be listed as **SAP**	
2.4	Next, select **request-response** for the **Exchange Pattern** field.	
2.5	For the **Type** field, select **IDoc**	
2.6	On the same line, next to the **Type** field, click on the **Select** button to search for the SAP object to be used in the transaction.	

2.7	Enter the following name in the **IDoc name** field: **DEBMAS03** and click on **Search.** Select the result that is returned and click on **Select**	
2.8	The **SAP Connector** provides a way to retrieve the XML and XSD for the SAP Object. Click on the **Details** button next to the **Object Name** field.	
2.9	In the **Details for DEBMAS03 IDoc** window, you can view and download the XML template and the XSD for the **DEBMAS03** IDoc.	

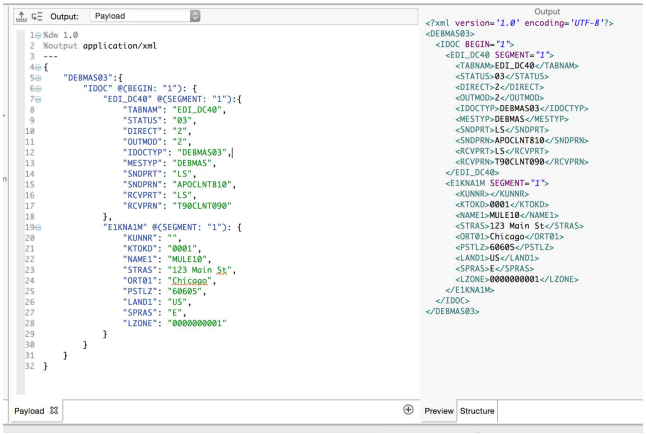
2.10	Click on **Export** and save the XML Template to the **src/main/resources** folder.	

3. Use DataWeave to Create an IDoc

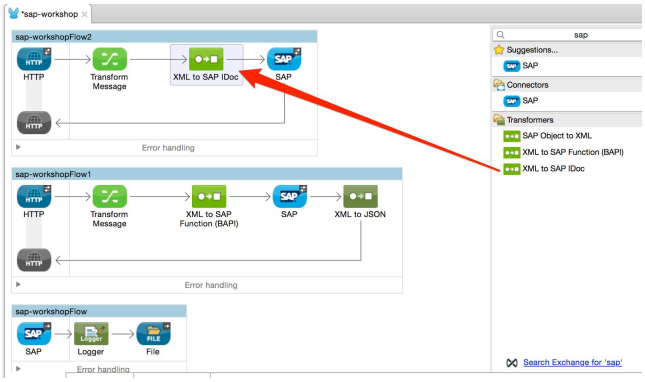
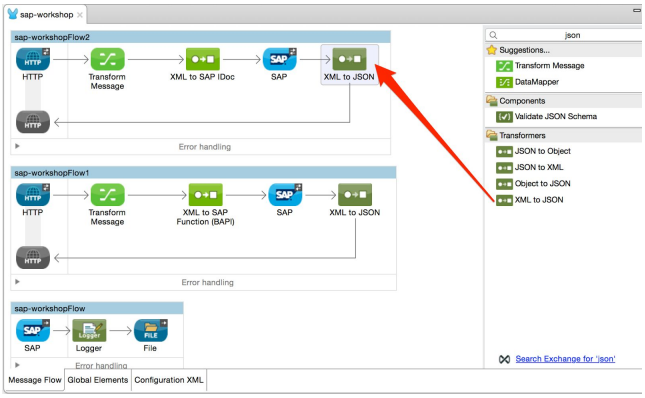
3.1	Now that the SAP Connector is configured with the correct credentials and SAP Object we intend to use, we need to setup the message to pass in. In the toolkit, search for the DataWeave component. Type in 'data' in the search and then drag and drop the **Transform Message** component into the canvas. Drop it between the **HTTP Connector** and the **SAP Connector**	
3.2	In the **Transform Message Configuration Properties** window, you can see how the component scaffolds out the expected message to be passed to the **SAP Connector** using **DataWeave** The **Output** on the right shows a preview of the payload message as you modify the message using DataWeave More information about the DataWeave language can be found here: http://mulesoft.github.io/data-weave	
3.3	Let's make some modifications to the message. Copy the following script paste over the existing script in the Transform section of the DataWeave component. Be sure the change the items highlighted in red to match your SAP instance settings and customer record that you would like to insert.	

```
%dw 1.0
%output application/xml
---
{
        "DEBMAS03":{
                "IDOC" @(BEGIN: "1"): {
                        "EDI_DC40" @(SEGMENT: "1"):{
                                "TABNAM": "EDI_DC40",
                                "STATUS": "03",
                                "DIRECT": "2",
                                "OUTMOD": "2",
                                "IDOCTYP": "DEBMAS03",
                                "MESTYP": "DEBMAS",
                                "SNDPRT": "LS",
                                "SNDPRN": "APOCLNT810",
                                "RCVPRT": "LS",
                                "RCVPRN": "T90CLNT090"
                        },
                        "E1KNA1M" @(SEGMENT: "1"): {
                                "KUNNR": "",
                                "KTOKD": "0001",
                                "NAME1": "MULE10",
                                "STRAS": "123 Main St",
                                "ORT01": "Chicago",
                                "PSTLZ": "60605",
                                "LAND1": "US",
                                "SPRAS": "E",
                                "LZONE": "0000000001"
                        }
                }
        }
}
```

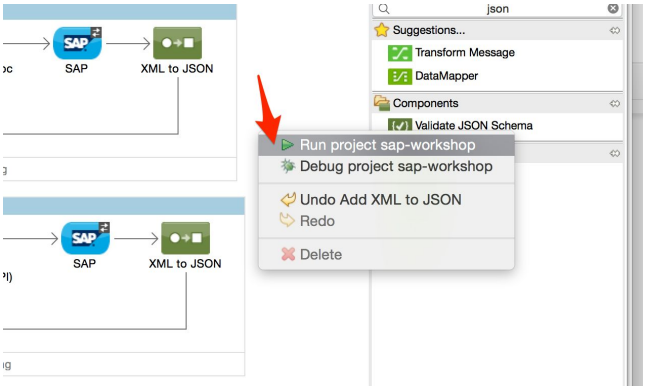
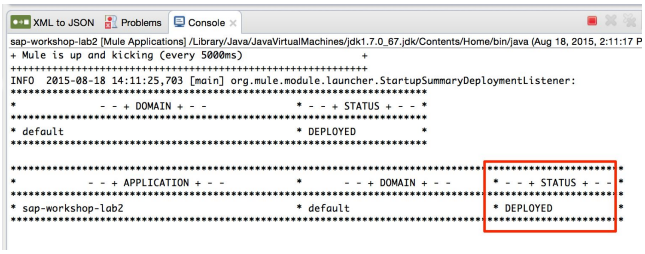
Notice how the preview on the right under **Output** updates the XML message that will be passed to SAP.

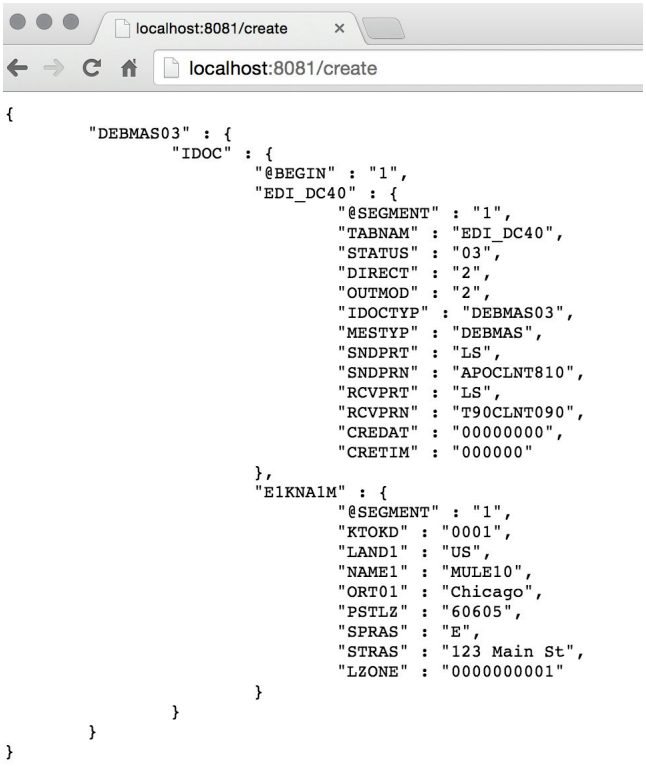
3.4	The script and output should look like the following screenshot.	

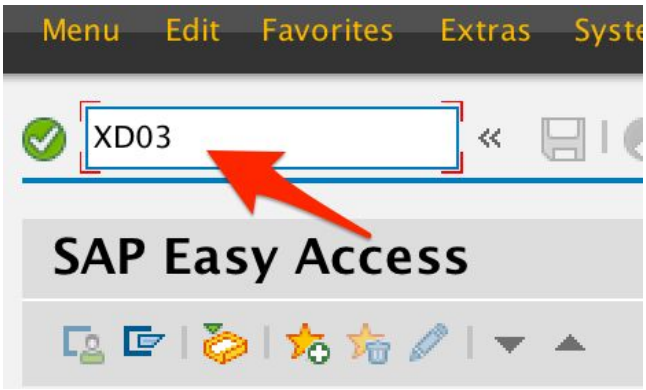
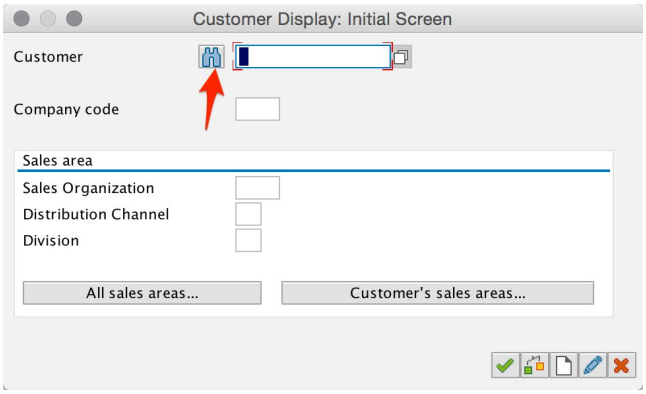
4. Transform XML to SAP IDoc

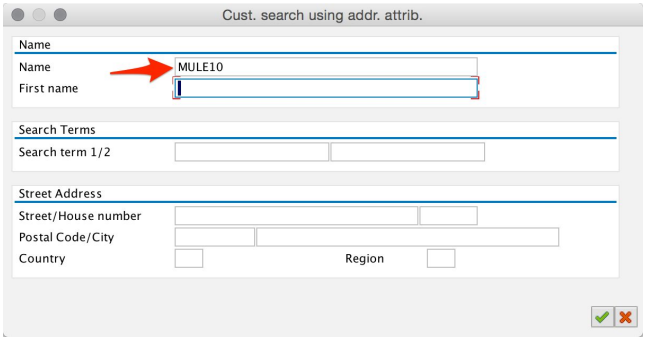
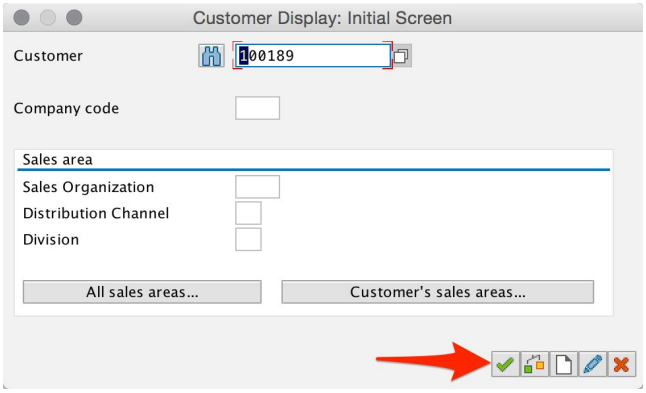
4.1	Now that the XML has the correct parameters, we need to transform it from XML to a SAP IDoc. From the toolkit, search for **SAP** and drag and drop the **XML to SAP IDoc** transformer and place it between **Transform Message** and the **SAP Connector**	
4.2	Once the SAP Connector processes the function the data will be returned as XML. Let's go ahead and transform that and output the customer list as JSON data. Search for **json** and then drag and drop the **XML to JSON** transformer and place it after the **SAP Connector**.	

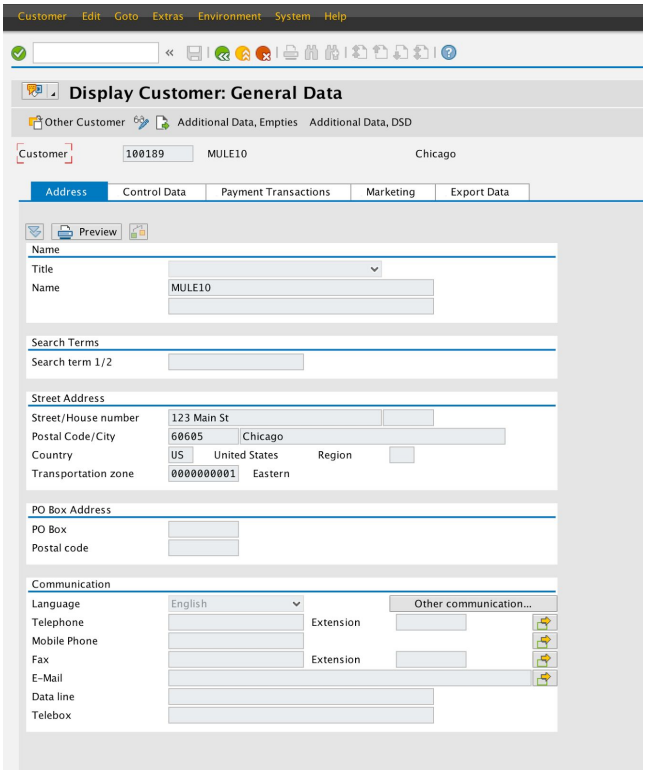
5. Test Project

5.1	Our next step is to test the flow we've built. Right-click on the canvas and select **Run project sap-workshop**	
5.2	The **Console** tab should pop-up now. Wait for the status to show **DEPLOYED** before moving onto the next step.	

5.3	Let's test out our flow now. Switch to your browser and enter the following URL:
	
	http://localhost:8081/create
	
	If everything was configured correctly, you should see the following screen on the right.
		
		{
		"DEBMAS03" : {
		"IDOC" : {
		"@BEGIN" : "1",
		"EDI_DC40" : {
		"@SEGMENT" : "1",
		"TABNAM" : "EDI_DC40",
		"STATUS" : "03",
		"DIRECT" : "2",
		"OUTMOD" : "2",
		"IDOCTYP" : "DEBMAS03",
		"MESTYP" : "DEBMAS",
		"SNDPRT" : "LS",
		"SNDPRN" : "APOCLNT810",
		"RCVPRT" : "LS",
		"RCVPRN" : "T90CLNT090",
		"CREDAT" : "00000000",
		"CRETIM" : "000000"
		},
		"E1KNA1M" : {
		"@SEGMENT" : "1",
		"KTOKD" : "0001",
		"LAND1" : "US",
		"NAME1" : "MULE10",
		"ORT01" : "Chicago",
		"PSTLZ" : "60605",
		"SPRAS" : "E",
		"STRAS" : "123 Main St",
		"LZONE" : "0000000001"
		}
		}
		}
		}

5.4	Switch over to SAP GUI and enter in transaction code **XD03** to find the customer that was inserted into SAP.	
5.5	In the **Customer Display: Initial Screen** pop-up window, click on the **Customer - Find**  button.	

5.6	In the **Name** field, enter the name of the customer that you passed over in the IDoc inside the node **E1KNA1M > NAME1**	
5.7	The **Hit List** window should show the newly created customer record. Select the customer record and click on the **Green** check mark.	
5.8	Back in the **Customer Display: Initial Screen** window, click on the **Green** check mark to continue.	

5.9	The customer that you inserted using an IDoc will be displayed if everything was configured successfully.	

Summary

This final lab demonstrated how to build an IDoc and insert a customer into SAP using the SAP Connector and Anypoint Studio.