
Before You Begin

Did you sign up for a special OmniStudio Developer Edition org already? You'll need one to do the steps in this guide. Here's how to request one if this is your first time completing an OmniStudio module:

1. Sign up for a [free Developer Edition org with OmniStudio](#).
2. Fill out the form.
 - a. For Email, enter an active email address.
 - b. For Username, enter a username that looks like an email address and is unique, but it doesn't need to be a valid email account (for example, [yourname@omnistudiotrails.com](#)).
 - c. After you fill out the form, click **Sign me up**. A confirmation message appears.
3. When you receive the activation email (this might take about 10 minutes), open it and click **Verify Account**.
4. Complete your registration by setting your password and security challenge question.

Tip: Write down your username, password, and login URL for easy access later.

You are logged in to your Developer Edition and you can begin practicing.

Build an OmniScript with Branching

Requirements

“As a service agent, I want a guided interaction for an account that provides me with options to:

- **Update the contact information** for the existing primary contact,
- **Change the primary contact** to another existing contact I can easily look up (and edit their contact information), or
- **Create a new contact** and assign them as the primary contact.”

Based on the user requirements, create an OmniScript that allows a user to update, change, or create a new primary contact.

Prerequisites

- None

Tasks

1. Edit an Update Account Primary Contact OmniScript
2. Add Conditional Branching to an OmniScript

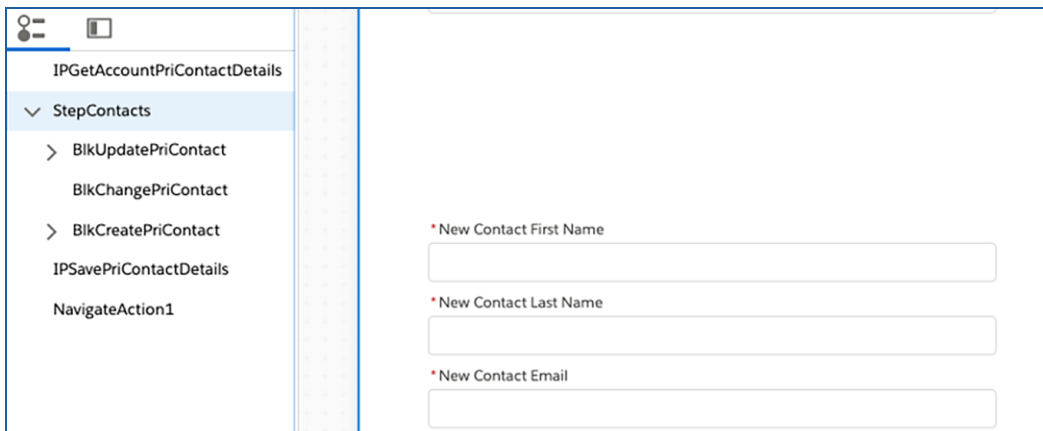
Time

- 30 mins
-

Task 1: Edit an Update Account Primary Contact OmniScript

1. Open the OmniScript **Team Starter Update Account Primary Contact**.
 - a. Click the **OmniScripts** tab.
 - b. In the Search box, type the keyword `team`.
 - c. From the search results, expand **team/updateAccountPrimaryContact**.
 - d. Click to open **Team Starter Update Account Primary Contact (Version 1)**.

The OmniScript opens in a new tab.
2. Create a new version of this OmniScript.
 - a. In the Header, click **New Version**. The new version opens in a new tab.
 - b. Close the tab with the previous version to prevent confusion later and continue in version 2.
 - c. Click **Edit**.
 - d. In the **Name** field, delete the word “Starter” to change the OmniScript name to **Team Update Account Primary Contact**.
 - e. Click **Save**.
3. Review the configuration as follows:
 - a. In the Canvas, select **IPGetAccountPriContactDetails** and review the **Properties** panel. This element contains an Integration Procedure called **team_getPrimaryContactDetails**, which gets data about a primary contact for a Salesforce account.
 - b. In the Navigation Panel, select the **Tree View** and expand **StepContacts**. Notice the **StepContacts** element has three distinct blocks: **BlkUpdatePriContact**, **BlkChangePriContact**, and **BlkCreatePriContact**.
 - c. On the canvas, click to expand **StepContacts**. The **BlkChangePriContact** is not visible there because it is empty.



4. Add some instructions to help guide the user through the interaction. It's best practice to ensure instructions are clear, concise, and complement the existing information in the UI.

- a. In the Canvas, click **StepContacts**.
- b. In the **Properties** panel, update the **Field Label** to say `Update Account Primary Contact`.
- c. Click the **Instruction** text box.
- d. Copy the following instructional text and paste it into the Edit Rich Text pop-up. You might need to copy and paste using CTRL + C and CTRL + V.

Update the contact information for the existing Primary Contact, change the Primary Contact and update their contact information, or create a new contact and make them the Primary Contact.



NOTE:
Copy the text or add instructions in your local language.

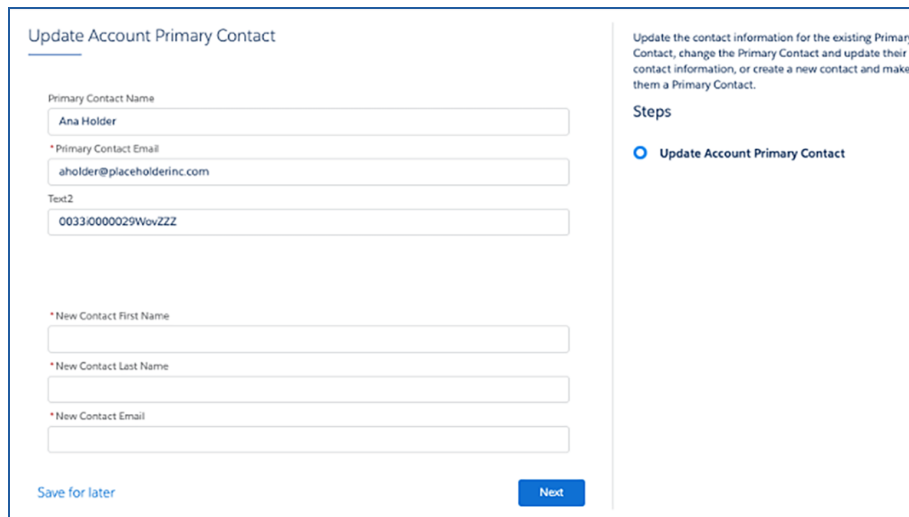
- e. Click **Save**.



NOTE:

If you leave the **Chart Label** field blank, the OmniScript displays the value in the **Field Label** field by default. You only need to use the **Chart Label** field if you wish to display an alternate label.

- f. In the Header, click **Preview** to view the instructions.
- g. Click the **x** to close the Data JSON.

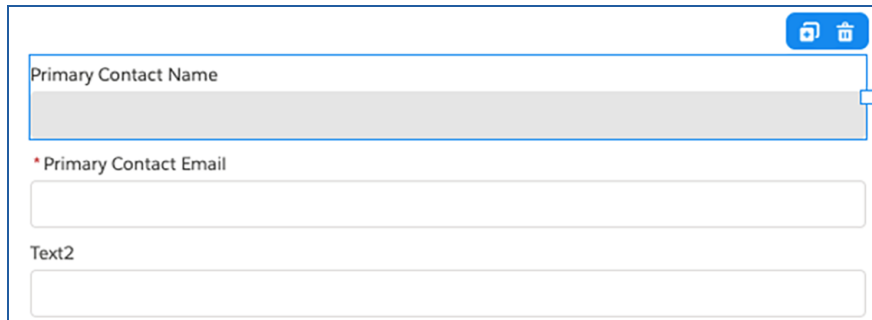


NOTE:

If you do not see the instructions, make sure Lightning is selected rather than Newport.

- 5. Make the Primary Contact Name field read-only. This field corresponds to an un-editable field in Salesforce, so it must be read-only.
 - a. In the Header, click **Design**.
 - b. In the StepContacts element, click the **UpdateName** text element. It is visible in the Canvas as **Primary Contact Name**.
 - c. In the **Properties** panel, select the **Read-only** checkbox.

Notice the field value is no longer editable because the Primary Contact Name is read-only.



The screenshot shows a form with three input fields. The first field is labeled "Primary Contact Name" and is currently empty. The second field is labeled "* Primary Contact Email" and is also empty. The third field is labeled "Text2" and is empty. There are lock and unlock icons in the top right corner of the form.




NOTE:

The Text2 field displays the ContactId, which is the RecordId for the primary contact. Keeping the RecordId in the same block of the OmniScript when you update that contact's details and save the changes back to Salesforce makes your life as a developer much easier. However, the RecordId has no purpose for the end-user. Therefore, hide the RecordId from the UI.

6. Configure the **BlkUpdatePriContact** block to hide the RecordId so that it is not visible in the UI.
 - a. In the Canvas, click the **UpdateContactId** text element (visible as **Text2**).
 - b. In the **Properties** panel, notice that there isn't a checkbox to hide the field. However, you can edit the JSON instead.
 - c. Scroll to the bottom of the **Properties** and click **Edit Properties As JSON**. The element JSON metadata defining this element is displayed.
 - d. Find the "hide" property and change the value to `true`. True/false Booleans must be all lowercase.

```
19   "conditionType": "Hide if False",
20   "accessibleInFutureSteps": false,
21   "debounceValue": 0,
22   "HTMLTemplateId": "",
23   "hide": true,
24   "disOnTplt": false,
25   "label": "Text2"
```

- e. At the bottom of the panel, click **Close JSON Editor**.
- f. In the header, click **Preview**.

- g. In the canvas, click the button to display the **Data JSON and Action Debugger**. 
 - h. Notice the RecordId is no longer visible in the UI. However, if you view the **Data JSON**, you see UpdateContactId. Also notice that StepContacts contains all the data to be saved.
 - i. In the Header, click **Design**.
7. View the Integration Procedure that saves the data.
- a. In the Canvas, click the **IPSavePriContactDetails** element. This is the Integration Procedure that saves the data for this element. It is configured to use the starter Integration Procedure: **team_savePrimaryContactDetails**.
 - b. In REMOTE PROPERTIES, under Extra Payload, notice the key value pair of **AccountId, %ContextId%**. This specifies that the OmniScript sends the ContextId on a variable named AccountId.
 - c. Expand SEND/RESPONSE TRANSFORMATIONS.
 - d. Rather than sending the entire JSON and identifying ContextId via extra payload, this OmniScript is sending just the JSON of the Step Account

Property	Value	Notes
Send JSON Path	StepContacts	This value is the node in the JSON of the OmniScript that you send to the Integration Procedure. In this case, you send the contents of the step.

NOTE:
Send JSON Node renames the node that you send.

The following properties trim the data returned to the OmniScript and rename it like the Send JSON Path and Send JSON Node in reverse:

- Response JSON Path
- Response JSON Node

Use the following properties to call a DataRaptor Transform for more complex data mapping when sending data or getting data back for the Integration Procedure:

- Pre-Transform DataRaptor Interface
- Post-Transform DataRaptor Interface

- e. Notice that the OmniScript sends just the **StepContacts** JSON path to the Integration Procedure.

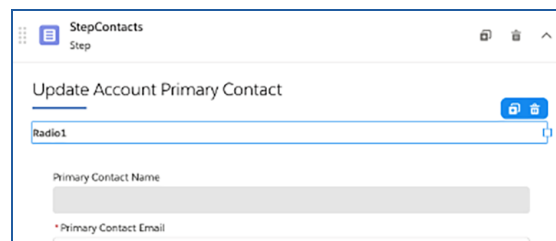
Review

Confirm your understanding by answering these questions.

1. Why not let a user edit the Name field from a Salesforce contact record?
2. Describe some of the ways to change the UI of OmniScript elements?
3. How is a Salesforce record uniquely identified?
4. What variable identifies the context of an OmniScript?

Task 2: Add Conditional Branching to an OmniScript

1. Create options that the user can select.
 - a. In the canvas, if it isn't already expanded, click to expand **StepContacts**
 - b. In the **Build** panel, expand the **INPUTS** section or use **Search** and enter `rad`.
 - c. Drag a **Radio** element to the Canvas at the top of the StepContacts element, as shown below. Be careful not to place the Radio element inside the top block instead!



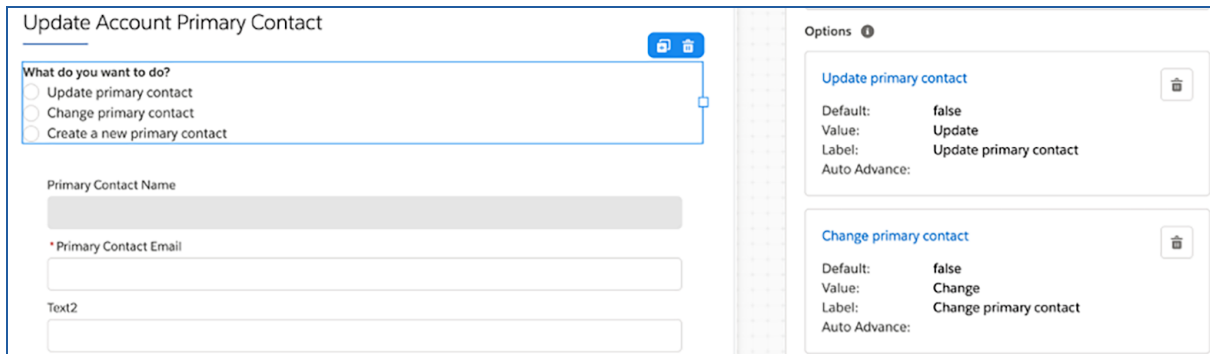
TIPS:

If you are having trouble adding an element to the OmniScript structure, check that you aren't zoomed out too much, as this sometimes prevents you from seeing the placement line when you're trying to drag and drop. (cmd + 0 or ctrl + 0)

- d. In the Radio element, change the Name to `RadioPriContact`.
- e. To help the end-user understand what to do next, for Field Label, add the following text:

What do you want to do?
- f. Keep **Display Mode** set to **Vertical**. Keep **Option Source** set to **Manual**. You are about to configure the options yourself in a way that is easier to read if they are presented vertically in a list.
- g. In the **Options** section, click **+ Add New Option** to add three options with the following values and labels for the associated variables in the OmniScript JSON. (You have to click **Save** to save the first one and then **+ Add New Option** to create the next one.)

Value	Label
Update	Update primary contact
Change	Change primary contact
Create	Create a new primary contact



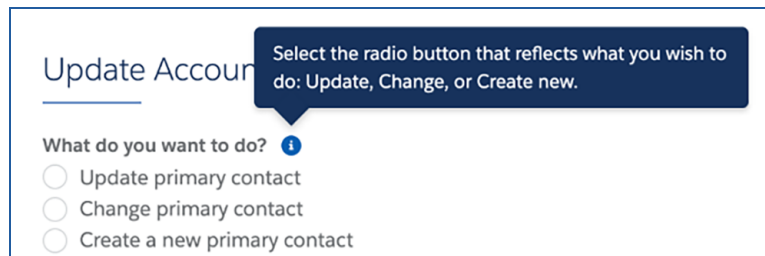
2. Add a help tooltip to help the end user.
 - a. Expand **HELP OPTIONS**.
 - b. Select **Activate Help Text**.
 - c. In **Help Text** enter the following: Select the radio button that reflects what you wish to do: Update, Change, or Create new.



TIPS:

Translation is available for help tooltips if multilanguage is enabled. For today, enter something in your local language if you prefer.

- d. In the Header, click **Preview**.
- e. Hover over the Tooltip icon next to **What do you want to do?** to view the tooltip.

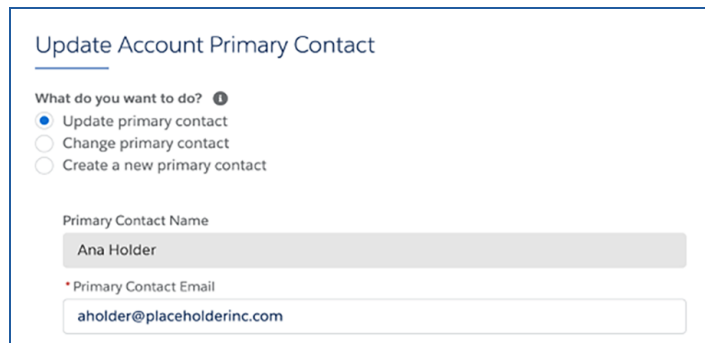


3. Configure a way for the value of the Radio element to control which block to display.
 - a. In the header, click **Design**.
 - b. Click the **BlkUpdatePriContact** element. (It is the block beneath the radio element.)
 - c. Notice **CONDITIONAL VIEW** at the bottom of the **Properties** panel. It allows you to set the conditions to **Show Element if True**.
 - d. Under **View Condition**, click **Show Element if True**.
 - e. Click in **Field** to open the pulldown list.
 - f. Choose **RadioPriContact**.
 - g. Verify that **Equal To** is selected.
 - h. For **Value**, type `Update`.
 - i. Click **Save**.
 - j. Notice the orange eye icon that appears indicating there is a condition configured for this block.
 - k. In the header, click **Preview**.
 - l. Notice that **BlkUpdatePriContact** is now missing from the UI.
 - m. Select the **Update primary contact** radio button. The block appears.



TIPS:

You might need to refresh here if this option is already selected.



4. Set the display condition for the next block in the step: **BlkChangePriContact**.
 - a. In the header, click **Design**.
 - b. In the Canvas, click **BlkChangePriContact**. (This is the empty block. You'll add elements to this block in a later exercise).
 - c. Under **CONDITIONAL VIEW > View Condition**, click **Show Element if True**.
 - d. For **Field** select **RadioPriContact**.
 - e. Confirm that **Equal To** is selected.
 - f. For **Value**, type `Change`.
 - g. Click **Save**.

5. Set the display condition for the last block in the step: **BlkCreatePriContact**.
 - a. In the Canvas, click **BlkCreatePriContact**.
 - b. Under **CONDITIONAL VIEW**, click **Show Element if True**.
 - c. For **Field** select **RadioPriContact**.
 - d. Confirm that **Equal To** is selected.
 - e. For **Value**, type `Create`.
 - f. Click **Save**.
 - g. Note all 3 blocks now have the orange eye icon in the upper-right corner indicating they have conditions.

- h. Click **Preview**. Because no options are selected, no fields are displayed. (If you see data, click to refresh).



BEST PRACTICE:

When designing guided interactions, consider showing options for the most common use case by default (if it does not result in end-users skipping a step). In this case, we've determined the most common use case is to update primary contact details.

6. Set the default display option to **Update primary contact**.
 - a. In the header, click **Design**.
 - b. In the canvas, click the **RadioPriContact** element.
 - c. In **Options**, click the **Update primary contact** link.
 - d. Select the **Use as Default Value** checkbox.
 - e. Click **Save**.
 - f. In the header, click **Preview**.
 - g. Notice the **Update primary contact** option is the default selection, and the primary contact data is populated.

Review

Confirm your understanding by answering these questions.

1. If an element has a conditional view, what setting do you need for that element to populate?
2. Which OmniScript elements allow users to make selections from a given set of options? What are the differences between these?
3. What are some best-practice guidelines for building OmniScripts with conditional views?