

Display External Data



NOTE:

Did you sign up for a [special OmniStudio Developer Edition org](#) already? You'll need one to do the steps in this guide. If not, use the link to fill out the form and have an org delivered to your inbox. The Exercise Guide in the first unit of this module has more detailed steps for this process if you need them.

Requirements

“As a service agent, I'd like to view weather forecast information for the account's location and see alerts if there are any hazardous weather conditions.”

You configure the OmniScript to display weather and forecast data from an external data source. You also configure an OmniStudio Action to launch the OmniScript from a FlexCard.

Prerequisites

- Build an OmniScript with Branching
- Create a Type Ahead Block Element
- Validate Data and Handle Errors

Tasks

1. Add a Weather FlexCard
2. Add a Weather Warning Banner

Time

- 20 mins

Task 1: Add the Current Weather

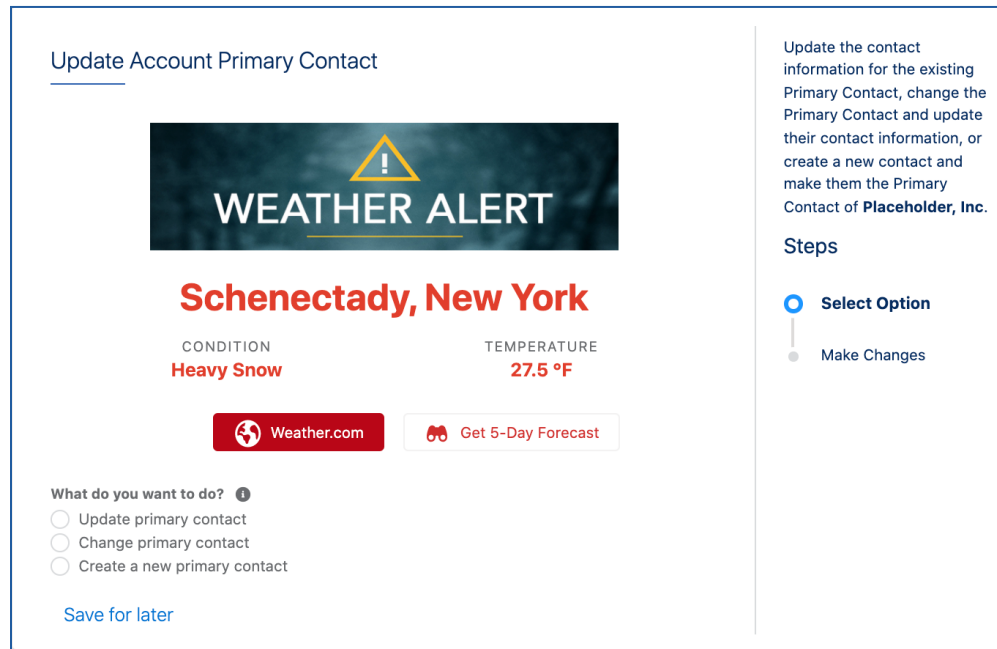
1. Create a new version of the **team/UpdateAccountPrimaryContact** OmniScript.
 - a. In the header, click **New Version**.
 - b. Close the tab with the previous version to prevent confusion later.
 - c. Click **Edit** and add `Weather` to the end of the name. The full name is now **Team Update Account Primary Contact Weather**.
 - d. Click **Save**.
2. Add an element that displays the weather FlexCard in the OmniScript.
 - a. In the header, click **Design**.
 - b. If it isn't already expanded, click to expand **StepRadio**.
 - c. In the **Build** panel, expand the **INPUTS** section.
 - d. Drag a **Custom Lightning Web Component** element into the top of the StepRadio step (above RadioPriContact).
 - e. In the **Properties**, for the element **Name**, use `WeatherCard`.
 - f. In **Lightning Web Component Name**, if you completed the FlexCard exercises, select **cfTeamWeather**. If you did not, select **cfSampleWeather**.
 - g. In **CUSTOM LIGHTNING WEB COMPONENT PROPERTIES** for **Property Name** enter `record-id` (the HTML attribute of recordId) and for **Property Source** enter `%ContextId%`.



NOTE: Custom LWCs do not display unless the OmniScript is active.

3. Preview the updates.
 - a. In the Header, click **Activate Version**.

- b. In the activation modal, when the OmniScript is active, click **Done**.
- c. Click **Preview**.
- d. The Weather card displays and is fully interactive.



Update Account Primary Contact

WEATHER ALERT

Schenectady, New York

CONDITION: **Heavy Snow** TEMPERATURE: **27.5 °F**

[Weather.com](#) [Get 5-Day Forecast](#)

What do you want to do? ⓘ

- Update primary contact
- Change primary contact
- Create a new primary contact

[Save for later](#)

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 of **Placeholder, Inc.**

Steps

- Select Option
- Make Changes

- e. In the header, click **Deactivate Version > Done > Design**.

Task 2: Add a Weather Warning Banner

1. Configure a second Integration Procedure to get weather data:
 - a. If needed, click **Deactivate Version** and return to **Design** view.
 - b. In the **Build** panel, expand the **ACTIONS** section.
 - c. Drag an **Integration Procedure Action** underneath the existing IPGetAccountPriContactDetails element and above the StepRadio element.
 - d. For the element **Name** and **Field Label**, use `IPGetWeatherDetails`.
 - e. From the **Integration Procedure** dropdown list, choose **team_getWeatherDetails**, which is a stub Integration Procedure.

You need to send two pieces of information to the Integration Procedure for when you have live data:

- i. The `ContextId`, which is already contained in the JSON.
 - ii. The number of days of forecast data, which is not present.
 - f. Expand **REMOTE PROPERTIES**.
 - g. Under **Extra Payload**, for the key, type `AccountId`, and for the value, type `%ContextId%`.
 - h. Click **+ Add New Key/Value Pair**.
 - i. For the key, type `Days`, and for the value, type `5` to specify a fixed value.
 - j. Click **Send Only Extra Payload** to only send the `ContextId` and days to the new Integration Procedure.
2. Preview and debug to review the response code:
 - a. In the header, click **Preview**.
 - b. Click **Action Debugger**.
 - c. Click **Clear Logs** to clear any existing data.

- d. In the canvas click **Refresh**. This sends the AccountId to the second Integration Procedure.
- e. Expand the **IPGetWeatherDetails** node and then the **Response** (you may have to expand the box in the lower-right corner to view more of it fully).



```
Response: {
  "IPResult": {
    "Forecast": [
      {
        "Date": "2019-01-26",
        "Condition": "Heavy Snow",
        "HiLoTempF": "30.5/21.7",
        "HiLoTempC": "-0.8/-5.7"
      },
      {
        "Date": "2019-01-27",
        "Condition": "Light Snow",
        "HiLoTempF": "31.7/22.8",
        "HiLoTempC": "-0.2/-5.1"
      },
      {
        "Date": "2019-01-28",
        "Condition": "Light Snow",
        "HiLoTempF": "31.6/18.7",
        "HiLoTempC": "-0.2/-5.2"
      }
    ]
  }
}
```

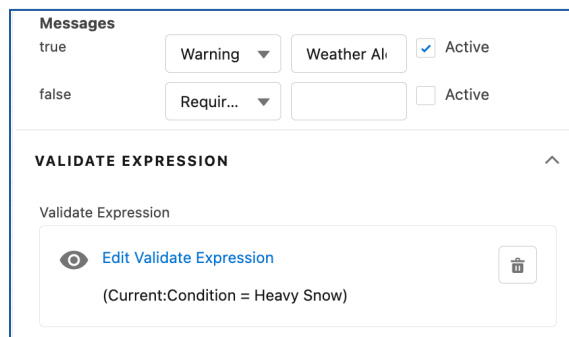
The response is JSON code with 2 nodes:

- i. A “Forecast” node, which has an array of data showing multiple forecast days. Because this is a stub Integration Procedure it always returns five days of data.
 - ii. A “Current” node, which has a set of sub-nodes providing data for temperature, condition, and city/state.
3. Add and configure a Messaging element with a logical condition so that the OmniScript shows a warning banner during hazardous weather conditions.
 - a. In the Header, click **Design**.
 - b. In the **Build** panel, expand the **FUNCTIONS** section.
 - c. Drag a **Messaging** element to the top of the **StepRadio** element (above the weather card).
 - d. For element **Name**, use `MsgWeatherAlert`.
 - e. Under Messages, for true, select **Warning** to display a Warning message if the condition is true.

- f. For the true Message, add the following merge field that shows the alert condition:

```
Weather Alert: %Current:Condition%
```


- g. For the false Message, uncheck Active because we are not using it. (You may not have issues with the stub data but will have problems when you add live data if you leave this checked.)
- h. Under **VALIDATE EXPRESSION**, click **Edit Validate Expression**.
- i. For the Field, specify that `Current:Condition` is **Equal To** Heavy Snow.
- j. Click **Save**.



The screenshot shows the configuration for two messages. The 'true' message has a 'Warning' type and the text 'Weather Al...'. It is active. The 'false' message has a 'Requir...' type and is inactive. Below the messages, there is a 'VALIDATE EXPRESSION' section with a 'Validate Expression' field containing '(Current:Condition = Heavy Snow)'. There is an 'Edit Validate Expression' button and a trash icon.

4. Preview the changes:
 - a. In the Header, click **Preview**.
 - b. Notice the yellow weather-warning banner is displayed, because the current condition in the stub data is **Heavy Snow** and because the OmniScript is no longer active, the Weather FlexCard is not visible.


Update Account Primary Contact

 Weather Alert: Heavy Snow

What do you want to do? ⓘ

- Update primary contact
- Change primary contact
- Create a new primary contact

[Save for later](#)

 **NOTE:** Custom LWCs do not display unless the OmniScript is active, so the Weather card will not display at this time.