# 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 the Current Weather
2. Add a Weather Warning Banner
3. Configure an OmniStudio Action for your OmniScript

## Time

- 50 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. Configure a second Integration Procedure to get weather data:

    a. Return to the Designer.

    b. In the **Build** panel, expand the **ACTIONS** section.

    c. Drag an **Integration Procedure Action** underneath the existing IPGetAccountPriContactDetails element and above the StepContacts element.

    d. For the element **Name** and **Field Label**, use `IPGetWeatherForecast`.

    e. From the **Integration Procedure** dropdown list, choose **team_getWeatherForecast**, 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 `2` to specify a fixed value.

   j. Click **Send Only Extra Payload** to only send the ContextId and days to the new Integration Procedure.

3. Preview and debug to review the response code:

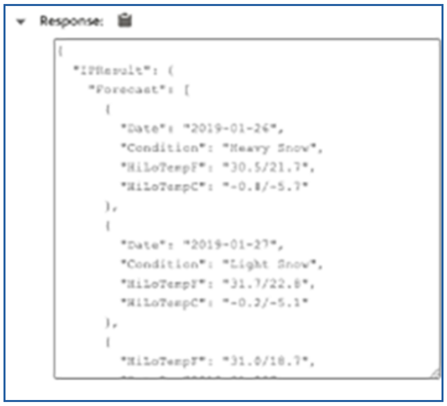
   a. In the header, click **Preview**.

   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 **IPGetWeatherForecast** 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"
      },
      {
        "HiLoTempF": "31.0/18.7",
```

   The response is JSON code with four 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.

iii. “`CurrentIcon`” and “`ForecastIcon`” nodes in the root with icon links. These link to an image on the API website, which you display in the OmniScript.

4. Add an element that displays the weather data in the OmniScript.

   a. In the header, click **Design**.

   b. In the **Build** panel, expand the **DISPLAY** section.

   c. If it isn’t already expanded, click to expand **StepContact**.

   d. Drag a **Text Block** element into the top of the StepContacts step (above RadioPriContact).

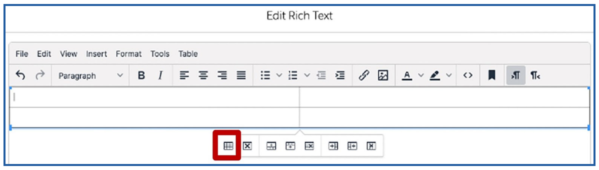
   e. In the **Properties**, for the element **Name**, use `TxtBlkWeather`.

**NOTE:**
A Text Block is a rich-text formatted HTML code block. Expand the editing workspace using the bottom-right down-arrow icon.

   f. Click in the **Text** box.

   g. From the **Table** dropdown menu, click **Table**, and add a 2 x 4 table.

   h. Click the Table Properties menu option.



   i. In **Width** enter `400px` and in **Height** enter `100px`.

   j. For **Border width** leave blank or enter `0`.

   k. Under **Advanced**, for **Border style** select **Hidden**.

   l. Click **Save**.

   m. Add the following text to the cells (use the temp format of your choice):

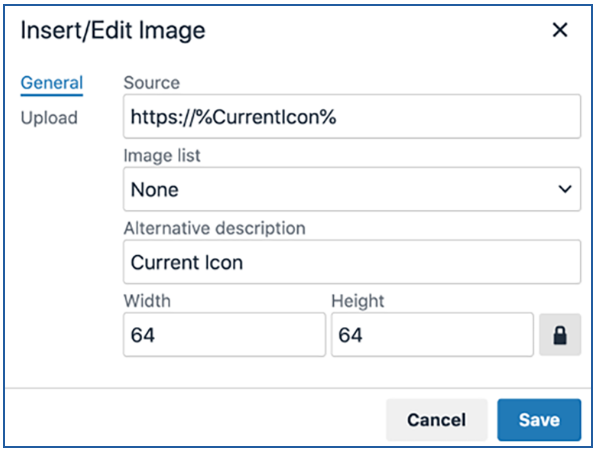
| | |
|---|---|
| `Current weather for:` | `%Current:CityState%` |
| `%Current:Condition%` | `%Current:TempF% °F`<br>`or`<br>`%Current:TempC% °C` |
| `Forecast weather for:` | `%Forecast|2:Date%` |
| `%Forecast|2:Condition%` | `%Forecast|2:HiLoTempF% °F`<br>`or`<br>`%Forecast|2:HiLoTempC% °C` |

To display the degree symbol: Mac – use **shift+option+8**, PC – use ASCII code 0176 (this is only typed using the numeric keypad, not the keyboard).

The merge code syntax for an array element is %parent|n:node% where n is the element number (starting at 1). For example, to display tomorrow's date (from the second element) you would use %Forecast|2:Date%

n. In the second row of the first column, place the cursor at the far left, click **Insert**, choose **Image....**, and set the following values in the modal:

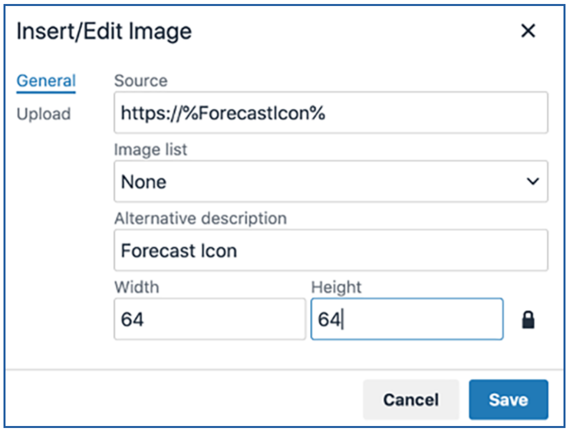
| **Property** | **Value** |
|---|---|
| Source | `https://%CurrentIcon%` |
| Alternative description | `Current Icon` |
| Dimensions | `64 x 64` |



Source: https://%CurrentIcon%
Image list: None
Alternative description: Current Icon
Width: 64
Height: 64

o. In the fourth row of the first column, place the cursor at the far left, click **Insert**, choose **Image....,** and set the following values in the modal:

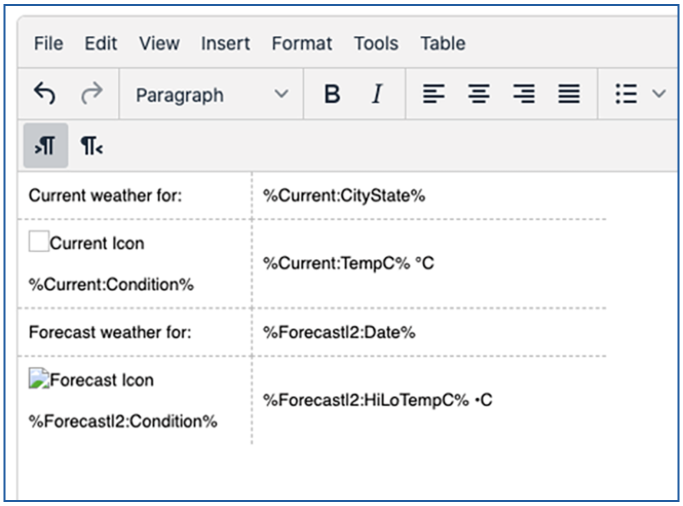
| Property | Value |
|---|---|
| Source | `https://%ForecastIcon%` |
| Alternative description | `Forecast Icon` |
| Dimensions | `64 x 64` |



Source: https://%ForecastIcon%
Image list: None
Alternative description: Forecast Icon
Width: 64
Height: 64

p. Click **Save**

q. Confirm the table matches the image below (with temp format of your choice):



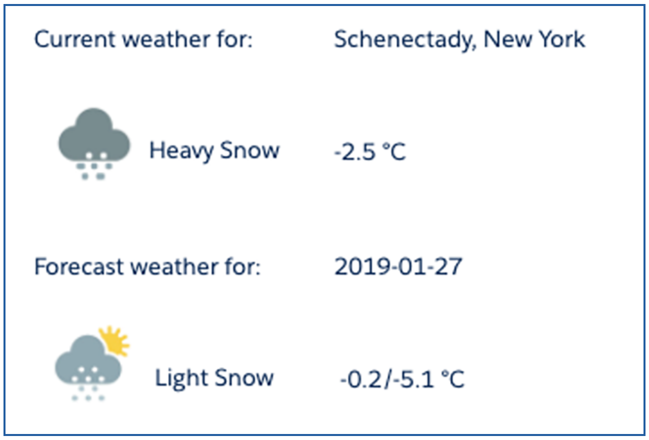
| | |
|---|---|
| Current weather for: | %Current:CityState% |
| □Current Icon<br>%Current:Condition% | %Current:TempC% °C |
| Forecast weather for: | %Forecastl2:Date% |
| Forecast Icon<br>%Forecastl2:Condition% | %Forecastl2:HiLoTempC% •C |

r. Click **Save**.

5. Preview the updates:

a. In the Header, click **Preview**.

b. Notice the weather conditions and temperature are displayed in the OmniScript UI.



| | |
|---|---|
| Current weather for: | Schenectady, New York |
| Heavy Snow | -2.5 °C |
| Forecast weather for: | 2019-01-27 |
| Light Snow | -0.2/-5.1 °C |

**NOTE:**
If the full table does not display, return to the Designer, Open **Rich Text Editor > View > Source Code** and remove and % values for **width** or **height** and return to the preview after saving the changes.

## Review

Confirm your understanding by answering these questions.

1. What are some examples of data that a service agent could find useful in a guided interaction that is not in Salesforce?
2. What is the merge field syntax for JSON sub nodes in OmniScripts?
3. Which OmniScript element is basically a rich-text formatted HTML code block?

## Task 2: Add a Weather Warning Banner

1. 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 **StepContacts** element (above the text block for the weather).

    d. For element **Name**, use `MsgWeatherAlert`.

    e. Under **CONDITIONAL VIEW > View Condition**, click **Show Element if True**.

    f. For the Field, specify that `Current:Condition` is **Equal To** `Heavy Snow`.

    g. Click **Save**.

    Now you need to specify the message shown on the banner.

    The following table shows the banner types available for true and false conditions.

| Message Type | Description | True | False |
|---|---|---|---|
| Comment | Grey banner | ✓ | ✓ |
| Success | Green banner | ✓ | ✓ |
| Warning | Yellow banner | ✓ | ✓ |
| Requirement | Red banner | | ✓ |

**ALERT:**
If you have Requirement for the active false statement, it means the element is required. The user cannot progress to the next step in the OmniScript until this condition is true.

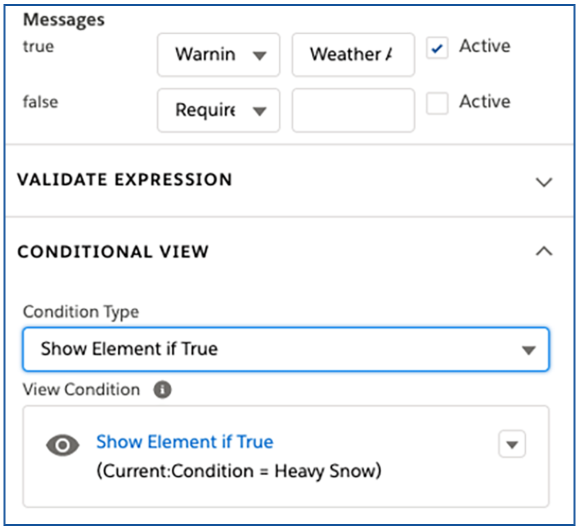
You can also deactivate either message using the options shown below.

h. Under Messages, for true, select **Warning** to display a Warning message if the condition is true.

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

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

j. 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.)



Messages

true Warnin ▾ Weather A ☑ Active

false Require ▾ ☐ Active

VALIDATE EXPRESSION

CONDITIONAL VIEW

Condition Type: Show Element if True ▾

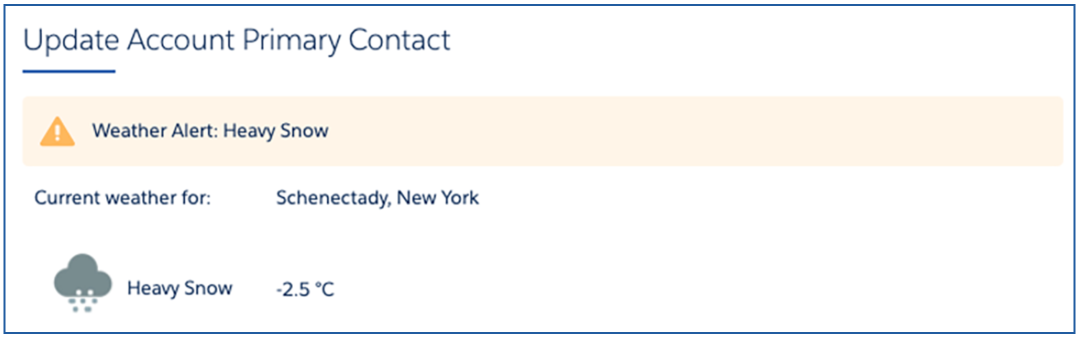
View Condition

Show Element if True
(Current:Condition = Heavy Snow)

2. 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**.



Update Account Primary Contact

⚠ Weather Alert: Heavy Snow

Current weather for: Schenectady, New York

Heavy Snow -2.5 °C

## Review

Confirm your understanding by answering these questions.

1. Which OmniScript element do you use to add a warning banner?
2. Which types/colors of notification banner are available for the OmniScript?
3. What is special about the Requirement message type? Why do you need to be careful when using it?

## Task 3: Configure an OmniStudio Action for your OmniScripts

1. Update the **team Update Primary Contact** OmniStudio Action details.

    a. Click the **App Launcher** to open the app menu.

    b. In the search field enter `Actions` to filter the options, then select **Vlocity Actions** to open the tab.

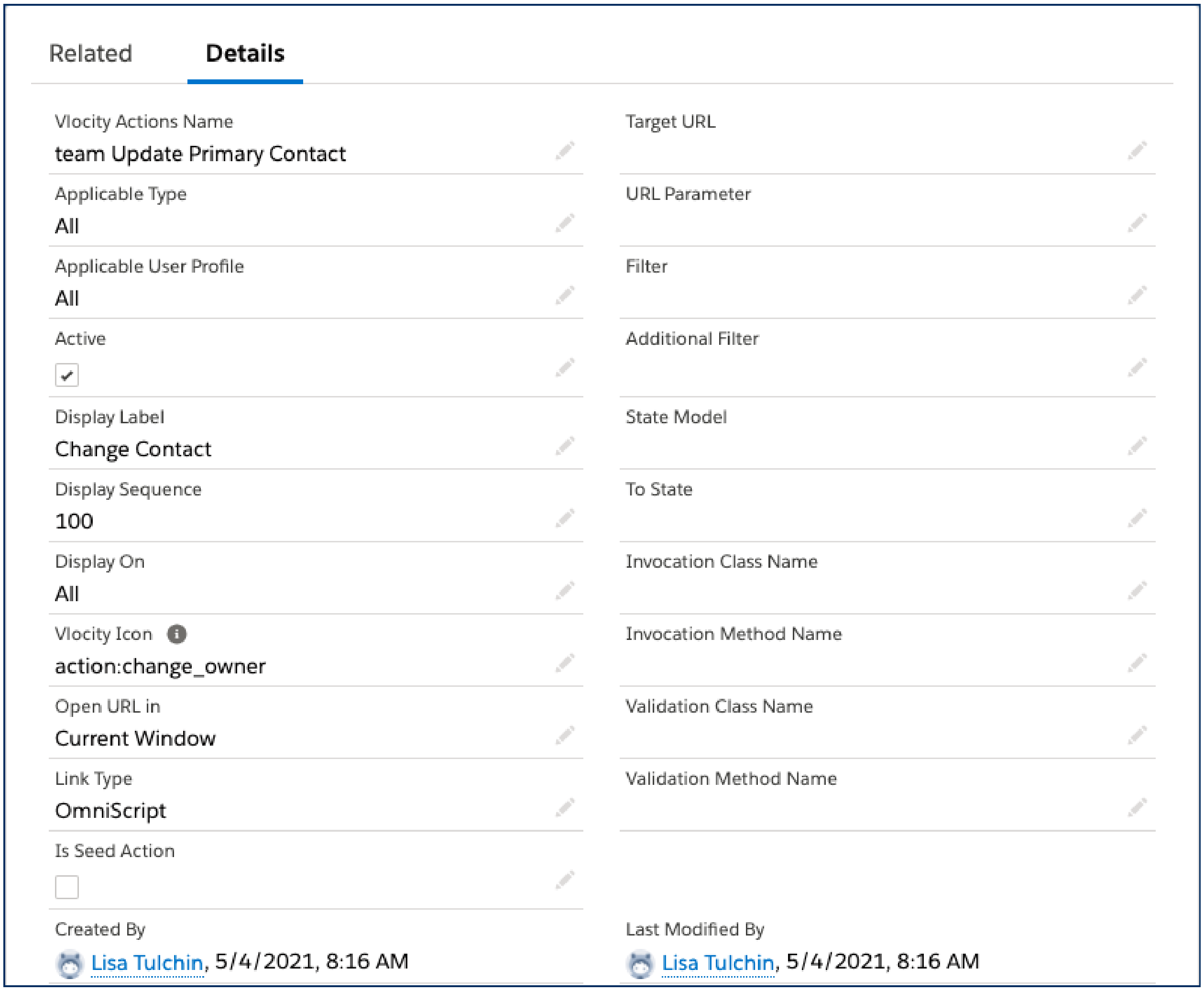
    c. If you do not see any Actions on the tab, change the view to **All** (The default in Salesforce is **Recently Viewed**).

    d. Open the **team Update Primary Contact** Action.

    e. Review the fields and notice the following:

| Field | Explanation/Definition |
|---|---|
| Applicable Type | The Objects the Action can be used with. (Best practice is to not use All because it slows down processing.) |
| Applicable User Profile | Which Salesforce User Profiles can see/use the Action. |
| Active | Like OmniScripts, FlexCards, or Integration Procedures, you cannot deploy an Action if it is not active. |
| Display On | Whether you can see the Action on Mobile, Web, or both. |
| Target URL | URL that the action launches, in our case an OmniScript. |
| URL Parameter | The field or fields that are being passed to the URL from the sObject named in the Salesforce Object Type. Usually this is Id, but it could be any field from the sObject. |

| | |
|---|---|
| Open URL in | Where the OmniScript opens. In a new tab / window or the same one. |
| Vlocity Icon | Indicates which icon is visible on a Card. You can view the full list of icons in the Edit modal below the details section. *These are only visible in Angular Cards*. |
| Link Type | What type of Action are you creating? OmniScript, ConsoleCards (used by the console), or Other |



Vlocity Actions Name
team Update Primary Contact

Applicable Type
All

Applicable User Profile
All

Active
☑

Display Label
Change Contact

Display Sequence
100

Display On
All

Vlocity Icon
action:change_owner

Open URL in
Current Window

Link Type
OmniScript

Is Seed Action
☐

Created By
Lisa Tulchin, 5/4/2021, 8:16 AM

Target URL

URL Parameter

Filter

Additional Filter

State Model

To State

Invocation Class Name

Invocation Method Name

Validation Class Name

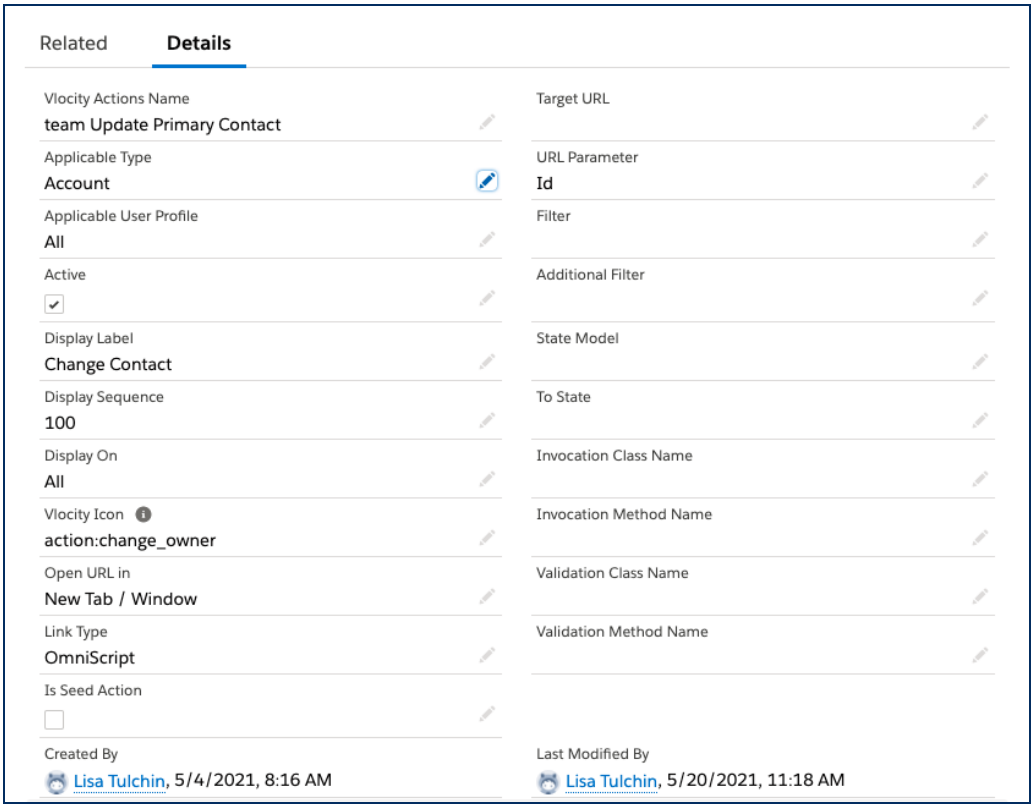
Validation Method Name

Last Modified By
Lisa Tulchin, 5/4/2021, 8:16 AM

f. Click **Edit**, and make the following edits to the Action (other fields remain as they are):

| Property | Value | Notes |
|---|---|---|
| Applicable Type | Account | Remove All and replace it with Account |
| URL Parameter | Id | |
| Display Sequence | 100 | |
| Open URL in | New Tab/Window | |
| Link Type | OmniScript | |

g. Click **Save**.



Vlocity Actions Name
team Update Primary Contact

Applicable Type
Account

Applicable User Profile
All

Active
☑

Display Label
Change Contact

Display Sequence
100

Display On
All

Vlocity Icon
action:change_owner

Open URL in
New Tab / Window

Link Type
OmniScript

Is Seed Action
□

Created By
Lisa Tulchin, 5/4/2021, 8:16 AM

Target URL

URL Parameter
Id

Filter

Additional Filter

State Model

To State

Invocation Class Name

Invocation Method Name

Validation Class Name

Validation Method Name

Last Modified By
Lisa Tulchin, 5/20/2021, 11:18 AM

2. Update the URL for the **team Update Primary Contact** OmniStudio Action using the **team Update Account Primary Contact** OmniScript.

    a. Return to the tab with **Team Update Account Primary Contact Weather**.

    b. In the header, click **Activate Version**. (You need to activate the OmniScript to use it elsewhere.)

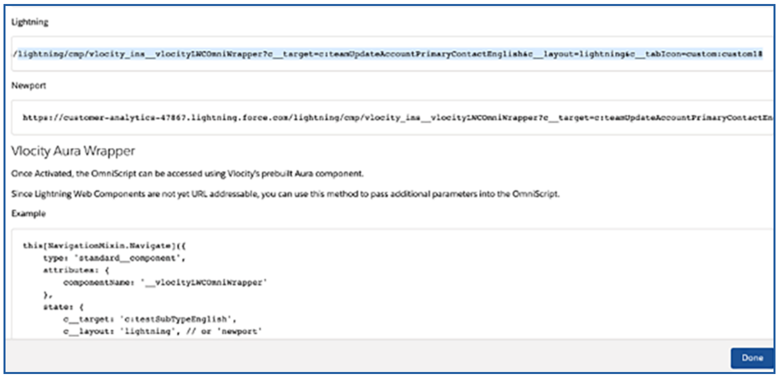
    c. When the chevrons are green, click **Done**.

    d. In the header, click the triangle next to **Edit** and select **How To Launch**.

    e. In the **Embedded** section, notice the tag that lets you embed the component into any Aura or LWC component.

    f. In the **Lightning** field, copy the URL from the / before **lightning** to the end. (If you wish, paste it where you are tracking items to copy and paste.)

    Copying the URL this way ensures portability for the OmniScript. If you copied the entire URL, you would only be able to deploy the OmniScript in your org.



Lightning

```
/lightning/cmp/vlocity_ins__vlocityLWCOmniWrapper?c__target=c:teamUpdateAccountPrimaryContactEnglish&c__layout=lightning&c__tabIcon=custom:custom18
```

Newport

```
https://customer-analytics-47867.lightning.force.com/lightning/cmp/vlocity_ins__vlocityLWCOmniWrapper?c__target=c:teamUpdateAccountPrimaryContactEn
```

Vlocity Aura Wrapper

Once Activated, the OmniScript can be accessed using Vlocity's prebuilt Aura component.

Since Lightning Web Components are not yet URL addressable, you can use this method to pass additional parameters into the OmniScript.

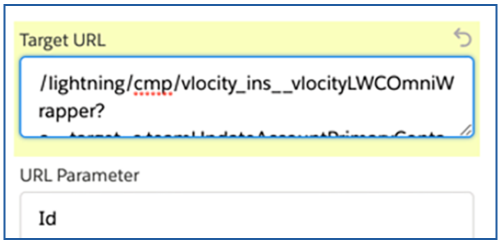
Example

```
this[NavigationMixin.Navigate]({
    type: 'standard__component',
    attributes: {
        componentName: '__vlocityLWCOmniWrapper'
    },
    state: {
        c__target: 'c:testSubTypeEnglish',
        c__layout: 'lightning', // or 'newport'
```

    g. Click **Done** to close the modal window.

    h. Return to the console tab with the **team Update Primary Contact** Action.

    i. Click in the **Target URL** field to open it up for editing and paste the URL you just copied into the field.



Target URL

/lightning/cmp/vlocity_ins__vlocityLWCOmniWrapper?

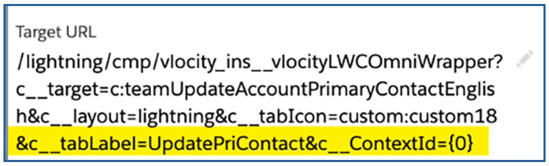
URL Parameter

Id

> **NOTE:**
> The example screenshot was taken in a training playground. Your training playground has a different URL.

j. Add `&c__tabLabel=UpdatePriContact&c__ContextId={0}` to the end of the URL. This adds a label for the tab that displays the OmniScript and passes the ContextId (the AccountId) to the OmniScript.

k. Click **Save**.



Target URL

/lightning/cmp/vlocity_ins__vlocityLWCOmniWrapper?c__target=c:teamUpdateAccountPrimaryContactEnglish&c__layout=lightning&c__tabIcon=custom:custom18&c__tabLabel=UpdatePriContact&c__ContextId={0}

## Review

Confirm your understanding by answering these questions.

1. What is an OmniStudio Action?
2. Can you deploy an Action if it is not active?
3. How did you build the URL to use in the Action?