TRACELINK UNIVERSITY

Home
Resources
TraceLink University

Custom XTT Link Actions Development Guide

Overview

This guide covers the steps necessary to create a custom Link Action. At the end, you will have a validated Javascript file and a custom transform. With these, you will work together with the TraceLink team on configuration.

1. Familiarize yourself with the relevant Link Action **terminology and principles**.
2. Use JavaScript to **make a direct API connection** between an external system and TraceLink to retrieve or send data.
3. **Create a custom transform** to map data fields between the two systems.
4. **Validate the Link Action** using TraceLink's XTT Link Actions Developer Kit.
5. **Upload** your Link Action for use.

Terminology and Principles

Each custom Link Action abides by the following principles:

- Connection is made via API to a single, external system
- Data is transmitted either inbound OR outbound
- Configured for a single TraceLink instance

- Configured for a single transaction type

Term	Definition
Canonical	TraceLink's system agnostic data format. The canonical data model allow information to be exchanged with any partner on the network, regardless of their data format.
Custom Transform	Data mapping of external systems with TraceLink canonicals.
Inbound Link Action	Facilitates transferring transaction data from an external system, via APIs, into TraceLink's canonical format.
Outbound Link Action	Facilitates transferring transaction data, via API, from TraceLink's canonical format into one needed for an external system.
Transaction Type	B2B messages relating to a business process (e.g. purchase orders, invoices, etc.).
Standard Link Actions	Link Actions built and maintained by TraceLink to connect to common, non-customized, third party systems.
Custom Link Actions	Link Actions built for external system not covered by Standard Link Actions or systems with customization.

Use the decision tree below to help you determine if you need a standard or custom Link Action.

* A list of external systems with Standard XTT Link Actions can be found here.

** A list of canonical fields for each transaction type can be found here.

Create a Custom Link Action

At this point you have determined that you need a custom Link Action. If the external system you want to connect to is not listed on this page, please contact us by emailing partner-operations [at] tracelink.com (partner-operations[at]tracelink[dot]com) before proceeding with this guide. We will need to work with you to configure it in the Link Action backend framework.

This step reviews how to create an inbound or outbound custom Link Action with JavaScript. The output will be a .js file that returns the JSON payload for the

designated transaction type.

Objects and methods available for both inbound and outbound Link Actions:

`linkActionContext` is the base object passed to provide configuration, rest methods, and debugging.

`linkActionContext.api.log` provides debug logging information:

```
linkActionContext.api.log(linkActionContext, "string to log");
```

`linkActionContext.api.oAuth2` provides OAuth2.0 authentication capabilities:

```
linkActionContext.api.oAuth2.get(LinkActionContext linkActionContext,
String url, String header, String payload);

linkActionContext.api.oAuth2.put(LinkActionContext linkActionContext,
String url, String header, String payload);

linkActionContext.api.oAuth2.post(LinkActionContext linkActionContext,
String url, String header, String payload);

linkActionContext.api.oAuth2.patch(LinkActionContext
linkActionContext, String url, String header, String payload);

// Return: {String url, String statusCode, Object header, String body,
String errorString}
```

`linkActionContext.config` provides configuration information, specific to a XTT Link Action's configuration in the framework. Locate the external system you are connecting to on this page for relevant configuration details. If the system is not listed, the TraceLink team will provide you with this information after configuration in the backend framework.

Inbound Link Actions

Inbound Link Actions fetch data from an external system, via API, and transform it

into TraceLink's canonical format. At a high level this entails:

- **Polling Data**: Check if there is new data since last poll
- **Querying the Records**: If there is new data, GET the records after the last poll time
- **Fetching Detailed Records**: Take the list of records from step two and GET detailed information about each one. This may require multiple API calls depending on how data is stored in the external system.
- **Processing**: Prepare the data into JSON format with all retrieved details

`cursor` is a stringified JSON object that stores the point when the external system's data was last sent inbound to OPUS. This typically is an epoch timestamp, but could be anything used to track a point in the data stream like an object ID.

`pollForDataInbound` is required for inbound link actions to get the given objects from the external system from the start point defined by the cursor.

```
function pollForDataInbound(linkActionContext, cursor) {
  // If cursor exists parse it
        cursor = parse(cursor)
  // Set up configurations (URL, record types, etc.)

  // GET purchase orders since last poll time
    purchaseOrders = queryPurchaseOrders(linkActionContext, cursor)
    //return purchase orders

    // Iterate through list of purchase orders and details about each
    for each order in purchaseOrders:
        detailedOrder = fetchPurchaseOrder(linkActionContext, order)

        // Convert into JSON
        postProcess(linkActionContext, cursor, detailedOrder)
}
```

A sample JavaScript file for inbound NetSuite purchase orders can be found here.

Now that you have your data from the external system in JSON format, you will need to create a custom transform to map its fields to the Tracelink canonical.

Outbound Link Actions

Outbound Link Actions fetch data in TraceLink's canonical format, apply a custom transform to it, and send the output via API to an external system. At a high level, this entails:

- **Preparing the Data**: Gather the data you want to send to an external system
- **Fetching Additional Identifiers**: Sometimes, the data you are sending needs more information, like unique IDs for certain items (e.g., the currency or terms used).
- **Sending the Data**: Send data to an external system via API
- **Handling the Response**: After sending the data, check if the operation was successful or if there was an error.

`sendDataOutbound` is required for outbound Link Action to make a PUT call with the given `outboundData` to send to the external system.

```
function sendDataOutbound(linkActionContext, outboundData) {

    // Create the URL for the outbound transaction
    const transactionType = 'purchaseorder';
    const uri = linkActionContext.config.url + '/' + transactionType;

    // Set the necessary headers for the API request
    const headers = { 'Content-Type': 'application/json' };

    // Parse the data you want to send
    let outboundDataJson = parse(outboundData);

    // Obtain any necessary identifiers
    outboundDataJson = fetchAndMergeIdentifiers(linkActionContext,
outboundDataJson);

    // Convert the updated outbound data to a string for sending
    const outboundDataString = JSON.stringify(outboundDataJson);

    // Send the outbound data to the external system using a POST
request
    let response =
```

```
linkActionContext.api.oAuth2.post(linkActionContext, uri, headers,
outboundDataString);

    // Check if the response is successful
    if (response.statusCode === '204') {
        return { success: true, message: "Data sent successfully" };
    } else {
        // Handle errors
        logError("Outbound failed: " + response.error);
        return { success: false, error: response.error };
    }
}
```

A sample JavaScript file for outbound NetSuite purchase orders can be found here.

Best Practices

We highly encourage you to create a generic, service account user in your external system to make the relevant API calls. This is because it is best practice to only assign the user access to the data it needs to function.

Creating a Custom Transform

Now that you have connected to an external system, you will need to transform the data using a custom transform. For inbound Link Actions, this means the data needs to be transformed from the external system's format into TraceLink's canonical format. For outbound Link Actions, this means data needs to be transformed from TraceLink's canonical format into the external system's format. At the end of this step you will have a custom transform file, .js or .jar, that maps data fields between the two systems.

You can find detailed implementation steps in our custom transform development guide.

Validating a Link Action

This step provides you with a tool to help validate your Link Actions JavaScript and custom transform are working properly.

This is done by setting up a local sandbox and using our XTT Link Actions Developer Kit. Detailed usage documentation can be found within our tooling guide.

If you encounter any issues with this tool, please email %20partner-operations [at] tracelink.com (partner-operations[at]tracelink[dot]com).

If you require a tool enhancement, please create an issue directly in the repository.

Now that you have validated your custom Link Action, you can upload it for use.

Upload and Use a Link Action

When you are ready to upload your validated custom Link Action for use, please send answers to the questions below to %20partner-operations [at] tracelink.com (partner-operations[at]tracelink[dot]com).

- What external system are you connecting to?
- Are you sending data inbound or outbound?
- If inbound, what is your desired polling frequency? This value must be greater than the Link Action execution time.

From here, you will receive follow up from the TraceLink team to work with you to:

- Assign the xtt-link-actions Enterprise Application to your TraceLink instance
- Assign appropriate user roles, detailed below
- Upload your Link Action JavaScript and custom transform
- Obtain required connection details such as client ID, client secret, URLs, etc.

Role	Permissions
Application Administrator	Add or Modify an XTT Link Action B2B connections
Member - Standard Access	View XTT Link Actions B2B connections Search and view B2B transactions
Link Action Developer	Can upload a Link Action to the Catalog
Link Action Administrator	Can promote XTT Link Actions from local to global

Change Management

Any changes made to the Link Action file or custom transform, require you to go through the upload steps above. Uploaded changes will overwrite the previous file(s), unless otherwise specified. If you have any questions about this process please contact partner-operations [at] tracelink.com (partner-operations[at]tracelink[dot]com).

Related Content

Link Actions

Pre-Built API connectors to popular enterprise software systems.
View More

Use Case: XTT Link Actions

Integrate with external systems using Link Actions.
View More

Standard Link Actions Development Guide

Availability and development practices for standard XTT Link Actions.
View More

XTT Link Actions Developer Kit

Configure your development environment.
View More