



SUITE ENGINE API ENGINE

A Suite Engine Application

MICROSOFT DYNAMICS 365 BUSINESS CENTRAL

Setup and User Guide

Version: 1.0

Last Revised: December 20, 2022



SETUP AND USER GUIDE

This document presents setup information regarding the API Engine functionality for Dynamics 365 Business Central.

Please ensure that the Suite Engine API Engine application and its dependent applications are correctly installed in your Microsoft Dynamics 365 Business Central instance before proceeding.

Contents

Introduction	1
Preliminary Setup.....	1
Managing Suite Engine Extensions	1
Creating and Managing Your Subscription	2
Understanding Endpoint URLs	4
The API Engine	5
Creating an API Set.....	6
Entering API Credentials	6
Exporting and Importing API Sets	7
Defining API Functions	7
Defining API Variables	8
Defining API Mappings.....	10
Copying API Functions.....	12
API Messages	13
Adding API Parameters to an API Message.....	14
Message Data Buffers	14

Introduction

The API engine allows you to build integrations from within Microsoft Dynamics 365 Business Central to external systems or platforms via API calls. It contains the tools necessary to define the structure, parameters, and variables of an API call, which can then be executed from within Business Central. Through the definition of mappings between external values and Business Central objects, retrieved data is recorded in the desired tables and fields.

The API Engine should be viewed as a framework that can be used to initiate and manage API calls from Business Central. It is not designed to service a particular industry or business process, but rather act as an agnostic architecture that can support any system that transmits data via API. This could range from a simple API that is built for a singular task such as retrieving weather information or calculating distance to something as complex as an eCommerce channel that requires the transmission of order, customer, and item data to and from the ERP platform.

The architecture of the API Engine is such that users without technical expertise can construct API calls through the user interface, with no coding required. Coding is supported, however, and developers can use the API Engine to address more advanced scenarios.

Preliminary Setup


Before you can start the process of working with the API Engine, there are some basic, preliminary steps you must perform.

Managing Suite Engine Extensions

The API Engine is comprised of multiple extensions:

- *Suite Engine Common Base*: this is used to manage your subscription to the API Engine and other Suite Engine products.
- *API Engine Library*: this is a library application that contains the API Engine functionality.
- *API Engine Library*: a “shell” application that facilitates communication between the API Engine Library and other systems.


It is necessary to enable communication to external systems for each of these extensions:

1. Choose the  icon, enter **Extension Management**, and then choose the related link.
2. On the **Extension Management** page, choose the *Suite Engine Common Base* extension.
3. Choose the **Configure** action on the **Manage** tab in the ribbon.
4. On the **Extension Settings** page, select the *Allow HttpClientRequests* check box.
5. Close the **Extensions Settings** page and repeat this process for the other API Engine extensions.

Creating and Managing Your Subscription

If the API Engine is your first Suite Engine product, it is necessary to create a new Suite Engine subscription. Suite Engine uses Stripe to manage subscriptions for its products.

You can set up your subscription in the **Suite Engine Subscriptions** page:

1. Choose the  icon, enter **Suite Engine Subscriptions**, and then choose the related link.
2. On the **Customer Information** FastTab, enter the following subscription information:
 - Name
 - Address
 - Address 2 (optional)
 - City
 - State
 - Postal Code
 - Country
 - Phone
 - Email
3. Choose the **Actions** ribbon, then choose the **Create Stripe Customer** action.

With the exception of **Address 2**, you must supply values in all of the fields on the **Suite Engine Subscriptions** page in order to create an account. If you attempt to create a new Stripe without populating one of these fields, an error message will inform you of what data you need to include. If all the required values are present, a new Suite Engine subscription account will be created and assigned in Business Central; the new subscription number will be displayed in the **Customer Id** field.

***Note:** while the subscription management functionality will validate whether all the required fields have been populated when a subscription is requested, it will not validate the information itself. Please be certain the values you enter in these fields are accurate.*

When you have created a new Suite Engine account, you can then request a trial subscription for the API Engine:

1. On the **My Products** FastTab, select the line for the API Engine.
2. Choose the **Manage** ribbon, then choose the **Create Free Trial Subscription** action. A subscription wizard will be presented to you.
3. On the first page of the wizard, place a check mark on the line for the subscription plan you want to enable (there should only be a single line) then choose the **Next** button.
4. On the subsequent wizard pages, validate your account details and subscription plan, choosing the **Next** button to advance.
5. On the final wizard page, choose the **Finish** button.

When you have completed the wizard, a new trial subscription for the API Engine is created under your account. The relevant subscription status fields on the **My Products** FastTab are updated to display the remaining length of the trial

subscription. Suite Engine products by default allow for a 30 day trial period, during which you have access to the full functionality of the application. If you wish to use API Engine following the trial period, you must activate your subscription by providing a valid payment method. You can supply a payment method and manage other aspects of your Suite Engine subscriptions through the Suite Engine Subscription Self Service Portal, which is accessed from the **Suite Engine Subscriptions** page by choosing the **Go To Payment Portal** action in the ribbon. This opens the Suite Engine Subscription Self Service Portal for the assigned Suite Engine customer ID. The portal presents information on the API Engine and any other Suite Engine solution subscriptions that have been set up under the same account; from here, you can add payment methods and adjust your existing subscriptions. The Suite Engine Subscription Self Service Portal is completely secure, and your payment information is safe.

***Note:** to minimize disruptions to your business, it is recommended that you visit the Suite Engine Subscription Self Service Portal and set up a payment method before your API Engine trial expires. You will not be charged until your trial period ends, at which point your subscription will be automatically activated.*

If you determine that the API Engine does not meet your business needs, you can cancel your subscription through the Suite Engine Subscription Self Service Portal at any time.

If the API Engine is being installed in a Business Central environment in which any of Suite Engine's other solutions have been deployed, the trial subscription will be created for the customer ID subscription account that is already assigned in Business Central. Although rare, there may be situations where a Suite Engine subscription account exists but no account details are present in Business Central; in such a scenario, it is necessary to link the Business Central environment to the appropriate subscription account from the **Suite Engine Subscriptions** page by choosing the **Link to Existing Customer** action on the **Actions** tab in the ribbon. In the **Link Existing Customer** page, you must enter the following information:

- The Suite Engine customer ID for the relevant subscription.
- The billing e-mail for the relevant subscription.
- The billing address (Address 1 only, information such as city and postal code is unnecessary) for the relevant subscription.

***Note:** if you do not know these values, please contact [Suite Engine support](#) for assistance.*

When this information is supplied, choose the **OK** button. The application will automatically assign the proper subscription account in Business Central and then update this account to include the API Engine.

After the initial entry of billing details on the **Suite Engine Subscriptions** page, these values become un-editable. If it is necessary to modify any of this information, you can do by choosing the **Edit Customer Information** action in the ribbon. This will open a separate window containing the existing customer values, which can then be edited. Changes that are made to this customer information will be automatically synchronized with the related Suite Engine subscription account.

It is also possible to make customer changes through the subscription licensing system and then synchronize these changes in Business Central from the **Suite Engine Subscriptions** page by choosing the **Force Refresh** action on the

Actions tab in the ribbon. This instructs the application to obtain updated subscription information from the subscription licensing system.

Understanding Endpoint URLs

A core concept of the API calls that the API Engine initiates from Business Central to an external system is the assignment of endpoint URLs to API functions. An endpoint URL is the link to which an API call will send and or/retrieve data. When a given API function is executed by the API Engine, the API call will be made to the endpoint that has been defined for that function.

The power of the API Engine comes in being able to create multiple API functions that represent different API calls to the external system. Each of these functions will require its own endpoint. In addition, many API functions are dynamic in nature, requiring endpoints that are equally dynamic. Accordingly, an important part of setting up the API Engine is the definition of the various endpoint URLs that will be used to initiate API calls from within Business Central.

There are three primary components of an endpoint URL:

- **Base URL:** this is base host or domain name to which API calls will be made.
- **Path:** further definition beyond the base URL as to where API calls will be made. A path can be comprised of multiple components, each one separated by a /. A path's components can be static values, or include variable identifiers.
- **Query Parameters:** different settings that further tailor the API call. Query parameters are appended to a URL path with a ? character, and multiple parameters are separated from one another by an &. There are different parameter types that can be added to a URL:
 - **Tags:** static values that represent fixed parameters. If the tag appears as part of a record, that record will be considered within the scope of the API call.
 - **Key Values:** parameters comprised of a key (a specific piece of information or condition) and a target value for that key. These two components are separated by an =.

To illustrate, consider the following example. We want to use the API Engine to retrieve jokes from the <https://v2.jokeapi.dev/> API. This API categorizes different types of jokes, and makes it possible to retrieve different jokes from these categories according to various parameters.

Here are four endpoint URLs we have created for this API:

1. <https://v2.jokeapi.dev/joke/Any>
2. <https://v2.jokeapi.dev/categories>
3. <https://v2.jokeapi.dev/joke/Programming?safe-mode&amount=10>
4. <https://v2.jokeapi.dev/joke/Any?safe-mode&amount=1>

Let's use these examples to explain the three components of an endpoint URL:

- **Base URL:** this is the base domain for a URL. As you can see, all of our URLs share the same base URL: <https://v2.jokeapi.dev/>.
- **Path:** these follow the base URL and direct the API call to different endpoints. In some cases, these paths are static. For example, URLs 1, 3, and 4 all have a /joke path, while URL 2 has a /categories URL. This indicates that we want to make an API call to the categories path with URL 2, and the joke path for the others. URL 3 also has a second static path: the /programming syntax. So in this case, we are instructing the API call to be made to the joke path, followed by the programming path.

URLS 1 and 4 both contain an /Any path following the fixed /joke value. This is an example of a variable URL path; the /Any instructs the API call to retrieve any data it obtains from the preceding path.

- **Query Parameters:** URLs 3 and 4 both include query parameters, as indicated by the ? and following text. The URLs contain one tag and one key value:
 - **Tag:** the safe-mode tag is included in both URLs. “Safe mode” is a tag that can be applied to the records (in this case, jokes) that are retrieved by the API call. By including this tag in the endpoint, we are instructing the API call to only retrieve jokes that have the safe mode designation.
 - **Key Value:** both URLs include the amount key, with different values. In this case, “amount” indicates the number of jokes we want to retrieve. For URL 3, we’ve indicated that we want to retrieve 10 jokes, while we want to retrieve 1 joke using URL 4.

If we re-visit our URLs, we should now be able to parse the syntax and understand their purpose:

1. <https://v2.jokeapi.dev/joke/Any> – retrieve jokes from any category, with no modifiers or conditions.
2. <https://v2.jokeapi.dev/categories> – retrieve joke categories.
3. <https://v2.jokeapi.dev/joke/Programming?safe-mode&amount=10> – retrieve 10 safe jokes from the programming category.
4. <https://v2.jokeapi.dev/joke/Any?safe-mode&amount=1> – retrieve 1 safe joke from any category.

The API Engine

Once you have set up your Suite Engine account and API Engine subscription, you are ready to start building API calls to external entities.

Because the API Engine exists as a development framework that you can use to build a wide variety of integrations, it is not possible to document all scenarios or configuration activities. The purpose of this documentation is to provide an overview of the different record types you need to set up to successfully make an API call for Business Central, as well as discussion of how these records can be configured in high-level scenarios. The basic understanding of the API Engine’s architecture should be your starting point for creating your own integrations.


More detailed instructions for specific use cases or more complex scenarios will be provided through supplementary documentation and other materials.

Creating an API Set

The various API calls that you want to make from Business Central to external entities are set up within the API Engine as “[API functions](#).” These API functions are then grouped together into what is called an “API set.” In this way, you can manage API functions by platform, channel, or other external system.

For example, suppose we wanted to integrate Business Central with external eCommerce, credit card payment, and rate shopping platforms. In this scenario, we would create a separate API set for each platform, then define the relevant API functions within each set.

Accordingly, when you want to build a new integration, the first step you must complete is the creation of an API set:

1. Choose the  icon, enter **API Sets**, and then choose the related link.
2. Choose a new line and enter a code that identifies the API set as unique in the **Code** field.
3. In the **Description** field, enter additional description for the API set.
4. Enter additional information about the external API, if desired:
 - **API Platform:** enter the platform on which the external API exists.
 - **API Version:** enter the external API’s version number/code.
 - **API Documentation URL:** enter a link to the external API’s documentation.

These fields exist for informational purposes only, and are optional.

Entering API Credentials

Once an API set is created, it is necessary to specify the necessary endpoint and credentials that will permit Business Central to make API requests to the external system. The specific requirements will differ from system to system; it may be that a very simple API requires just a single endpoint, while something like an eCommerce channel requires the entry for credentials that are obtained from the platform. For this reason, the API engine’s credentialing functionality is built to accommodate a wide range of credentialing authorization types.

To enter API credentials:

1. With the relevant API set selected on the **API Sets** page, choose the **API Credentials** action in the ribbon.
2. On the **API Credentials** page, choose the **New** action in the ribbon to open a new API credential card.
3. In the **Description** field, enter additional description for the API set’s credentials.
4. In the **Authorization Type** field, choose the type of authorization used by the API. The following authorization types are supported:
 - Basic Authentication
 - Bearer Token
 - API Key
 - OAuth 2.0
 - AWS Signature

In addition, you can also indicate that no authorization is required.

5. In the **API Endpoint URL** field, enter the URL to which all functions contained within the API set will use. It is possible to perform additional endpoint definition for specific API functions, but it is still necessary to define a base URL against which all of these variables will be enacted. If we consider the [components of an endpoint URL](#), the base URL would generally be assigned at the credential level.
6. The other FastTabs on this page support the different authentication types. Choose the FastTab that corresponds to the selection in the **Authorization Type** field and fill out the required credential values.

Exporting and Importing API Sets

You can export and import API sets and their underlying components from within Business Central. This is useful if you build an API set that you want to use in another environment. It is also helpful if you want to create a new API set that is largely similar to an existing API set; rather than start from scratch, you can export the existing API set, then make any necessary changes before importing it back into Business Central.

To export an API set:

1. With the relevant API set selected on the **API Sets** page, choose the **Export** action in the ribbon.
2. On the **Export Data** FastTab of the **Export API Configuration** page, choose the API set records that you want to include as part of the export.
3. Choose the **Export** action in the ribbon.

The API Engine will export the API set's data to a JSON-formatted text file. This file can then be imported into another Business Central environment (or modified and then imported back into the original Business Central environment).

To import an API set:

1. On the **API Sets** page, choose the **Import** action in the ribbon.
2. Navigate to the location of the file you want to import, then select the file and choose the **Open** button.

The API Engine will import the selected file's data and create a new API set, along with any underlying records.

Defining API Functions

Once you have set up an API set and established its endpoint and any necessary credentials, you can create the functions that are containers within the set. Separate API function records are defined for the different API calls you want to make from Business Central. For example, if we were integrating to an external platform and wanted to send or retrieve standard master data records, we might create different API functions for sending customers, sending vendors, sending items, and so forth.

To define API functions:

1. With the relevant API set selected on the **API Sets** page, choose the **API Functions** action in the ribbon.
2. Choose a new line and enter a code that identifies the API function as unique in the **Function Code** field.
3. In the **Description** field, enter additional description for the API function.

By default, the **Request Processing Method** field contains a single *Default* option. The API Engine provides the capability for developers to create custom methods via codeunit interfaces, however, so it is possible for additional options to be available. The process of creating custom request processing methods is explained in other Suite Engine documentation.

4. In the **Static Path** field, enter any static path components of the API function's endpoint URL. This syntax will be appended to the API set's endpoint URL that you assigned to the API set's [credentials](#). You can also use [API variables](#) to assign path values to an API function's endpoint.
5. In the **Buffer Processing Type** field, indicate the structure in which [data buffers](#) for API messages that are created from the function are created.
6. In the **Response Processing Method** field, indicate the manner in which the API Engine should process responses to the API function's calls:
 - *None*: API calls will not be processed by the API Engine.
 - *Single Record Update*: the API Engine will create or update a single record in Business Central.
 - *Multiple Records Update*: the API Engine will create or update multiple records in Business Central.
7. When you have finished configuring the API function, you can test it by choosing the **Execute** action in the ribbon. A notification will ask if you want to open the new [API message](#) that was created.
8. Choose the **Yes** button to review this API message and confirm that the API function is making API calls as intended.

Defining API Variables

When you are initially setting up an API set, it is necessary to define a single endpoint URL that all of that API set's functions will use when making API calls. However, every API function that is part of an API set exists for a different purpose, and it is unlikely that they will all use the same endpoint. The API Engine provides you with the ability to define variables that will augment or override an API set's base URL on a function by function basis.

URL Parameters

URL parameters are settings that influence the nature of an API call. Parameters can be established to refine the type or volume of data that is requested of the external system. For example, if you wanted a particular API function to focus on a particular category or status of record type, you would append the necessary parameter to its endpoint.

To define URL parameters:

1. With the relevant API function selected on the **API Functions** page, choose the **API Variables** ribbon, then choose the **URL Parameters** action.
2. Choose a new line and enter a check mark in the **Auto Build Request** field. This will instruct the API Engine to automatically include the variable as part of API requests to the external system.
3. By default a value will be assigned in the **Auto Build Sequence** field, but you can change it, if desired. If you have multiple API variable lines, you can use this field to instruct the API Engine how to sequence them as part of API requests to the external system.
4. In the **Name** field, enter a name to identify the API parameter

5. In the **Variable Value Type** field, choose the type of parameter you want to create. The selection you make in this field depends on the type of parameter you want to define:
 - If you want to define a tag, choose the *Text Value* option.
 - If you want to define a key value, choose the *Key Value Text* option.
6. In the **Value Processing Type** field, choose the type of parameter you want to create:
 - *Static*: The parameter will be entered as a static value.
 - *Parameter Record*: a variable parameter that is defined at runtime.
7. The selected variable processing type will determine the remainder of your setup activities:
 - If the value processing type is *Static*:
 - Enter the parameter value in the **Static Value** field. This will be included as part of the endpoint URL, so this syntax must adhere to the external system's API guidelines.
 - In you are defining a key tag, you must also enter the key in the **Static Name** field. This will be included as part of the endpoint URL, so this syntax must adhere to the external system's API guidelines.
 - If the value processing type is *Parameter Record*:
 - Enter or use the AssistButton in the **Value Table No.** field to assign a table to be used in defining the variable path. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
 - Enter or use the AssistButton in the **Value Field No.** field to assign a field from the specified value table to be used in defining the variable path. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
8. In the **Variable Value Null Behavior** field, use the dropdown to instruct the API Engine how to handle variable requests that are cannot be resolved during the API call.

URL Paths

Because separate API functions exist for different API calls, it is necessary to define unique endpoint URLs for each one. It is possible to define static path values for an API function in the **API Function** table, but for variable path values, it is necessary to use API variables. You can also use API variables for static path values, if desired.

To define URL paths:

1. With the relevant API function selected on the **API Functions** page, choose the **API Variables** ribbon, then choose the **URL Paths** action.
2. Choose a new line and enter a check mark in the **Auto Build Request** field. This will instruct the API Engine to automatically include the variable as part of API requests to the external system.
3. By default a value will be assigned in the **Auto Build Sequence** field, but you can change it, if desired. If you have multiple API variable lines, you can use this field to instruct the API Engine how to sequence them as part of API requests to the external system.
4. In the **Name** field, enter a name to identify the API variable.
5. In the **Variable Value Type** field, choose the *Text Value* option. URL paths are always entered as text.
6. In the **Value Processing Type** field, choose the type of path value you want to create:
 - *Static*: The path will be entered as a static value.

- *Parameter Record*: a variable path that is defined at runtime according to one or more parameters.
7. The selected variable processing type will determine the remainder of your setup activities:
 - If the value processing type is *Static*, enter the path value in the **Static Value** field. This will be included as part of the endpoint URL, so this syntax must adhere to the external system's API guidelines.
 - If the value processing type is *Parameter Record*:
 - Enter or use the AssistButton in the **Value Table No.** field to assign a table to be used in defining the variable path. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
 - Enter or use the AssistButton in the **Value Field No.** field to assign a field from the specified value table to be used in defining the variable path. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
 8. In the **Variable Value Null Behavior** field, use the dropdown to instruct the API Engine how to handle variable requests that are cannot be resolved during the API call.

URL Overrides

It is possible to completely override an API set's endpoint for a given API function. When an API function is executed, any URL overrides that have been defined for that function will be used for the related API call. While not as dynamic as the other API variable options, this can be a quick and easy way to configure an API function's endpoint if a high level of variability to not necessary.

To define URL overrides:

1. With the relevant API function selected on the **API Functions** page, choose the **API Variables** ribbon, then choose the **URL Overrides** action.
2. Choose a new line and enter a check mark in the **Auto Build Request** field. This will instruct the API Engine to automatically include the variable as part of API requests to the external system.
3. By default a value will be assigned in the **Auto Build Sequence** field, but you can change it, if desired. If you have multiple API variable lines, you can use this field to instruct the API Engine how to sequence them as part of API requests to the external system.
4. In the **Name** field, enter a name to identify the API variable.
5. In the **Variable Value Type** field, choose the *Text Value* option. Override URLs are always entered as text.
6. In the **Value Processing Type** field, choose the *Static* option.
7. In the **Static Value** field, enter the URL you want the related API function to use when making API calls to the external system.

The URL override setup is complete. The next time the API function is executed, the API call will be made to this override value, rather than the related API set's default endpoint.

Defining API Mappings

When data is retrieved into Business Central from an external platform as part of an API call, it is necessary instruct the system how to process this data. The API Engine provides you with the ability to define mappings between the data that

an API function retrieves from an external system and corresponding objects in Business Central. In this way, retrieved data can be entered and processed according to Business Central's standard functionality.

For example, suppose we were creating an API integration between Business Central and an external product information management (PIM) system. As part of this integration, we define an API function to retrieve product information from the PIM system into Business Central and then create a new item record. In this scenario, we would define API mappings between the data we retrieved from the external system and fields in the **Item** table. When the API function retrieves information, a new item record is created and its values are populated according to these mappings.

You can view an API function's API mappings from the **API Functions** page by selecting the relevant API function line, then choosing the **API Mappings** action in the ribbon. You can manually define API mappings for an API function, but a more efficient way of performing this setup activity is to first generate an API message for the API function, then using that message's [data buffer](#) as the basis for the function's API mapping definition:

1. After [configuring an API function](#) on the **API Functions** page, choose the **Execute** action in the ribbon. A notification will ask if you want to open the new API message that was created.
2. Choose the **Yes** button to open the API message.
3. Choose the **Related** ribbon, then choose the **Message Data Buffer** action.
4. On the **API Data Buffer** page, choose the **Update API Mapping** action in the ribbon.

The API Engine will use the API message's data buffer to build mappings for the related API function.

5. Close the API data buffer and API message to return to the **API Functions** page.
6. Choose the relevant API message, then choose the **API Mappings** action in the ribbon.
7. The first API mapping line that is created from the data buffer is the top-level process that serves as the root parent to all subsequent mappings, and should be edited as follows:
 - Place a check mark in the **Function Root Node** field to identify this mapping as the root process.
 - In the **Source Type** field, choose the *TableNode* option.
 - Enter or use the AssistButton in the **Table No.** field to assign the table to which all API mappings that exist within this root mapping will write their data. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
8. You then must indicate which API mapping line represents the starting point for a record's mappings. As the API Engine processes an API call's response, this API mapping will instruct it to create a new record in the specified table. Subsequent mappings will enter data against that record.
 - In the **Source Type** field, choose the *RecordNode* option.
 - Enter or use the AssistButton in the **Table No.** field to assign the table to which all API mappings that exist within this record will write their data. This could be a standard Business Central table, or a custom one that has been created for a specific organization. Note that it is possible to build API mapping structures that feature record nodes within record nodes, allowing for a single API function to enter data in a given table as well as one or more sub-tables.
 - If the API function retrieves data as a single array enter or use the AssistButton in the **Field No.** field to assign the field from the specified table to which you want to map the relevant piece of information

represented by the API mapping line. If the data is not retrieved as part of a single array but rather through multiple elements, you will perform this field mapping for the API mapping lines that represent these elements, as described in the next step.

9. Once you have defined the table and record nodes, it is then necessary for you to map the different values that are retrieved as part of the API call's response to corresponding fields in Business Central. On each line that you want to map to a Business central field:
 - In the **Source Type** field, choose the *Element* option.
 - Enter or use the AssistButton in the **Table No.** field to assign the table to which you want to map the relevant piece of information represented by the API mapping line. This could be a standard Business Central table, or a custom one that has been created for a specific organization.
 - Enter or use the AssistButton in the **Field No.** field to assign the field from the specified table to which you want to map the relevant piece of information represented by the API mapping line.

If the selected field is a part of the assigned table's primary key, the **IsKeyField** field will be automatically checked by the API Engine.

You can also manually enter API mappings, if desired. This is especially useful if it is necessary to create a mapping beyond the scope of what the API Engine can generate from an API message's data buffer. In addition to the Table Node, Record Node, and Element source types described above, you can create mappings for the following source types:

Record Node Value: the API mapping is included as part of the record, as identified by the Record Node mapping, but does not have a dedicated element as retrieved from the external system. This is useful in scenarios where you retrieve a single value from the external platform and want to enter it in a Business Central table that requires a code and description. In such a scenario, you would map the Record Node level line to the table's **Code** field, then manually create a separate Record Node Value line that maps the same piece of data to the table's **Description** field.

When you are manually entering API mapping lines, you must assign a value in the **Sequence** field to indicate how it should be processed in relation to other mappings. The API mapping lines that are automatically created from a message's data buffer are incremented in values of 10, making it easy to insert a new API mapping line wherever desired.

Copying API Functions

It is possible to select an existing API function and create a copy of it as a new record. This can expedite setting up new API functions that share many of the same values and parameters as an existing function; in such a scenario, you can copy the existing function to a new record, then make any necessary modifications.

To copy an API function:

1. With the relevant API function selected on the **API Functions** page, choose the **Actions** ribbon, then the **Copy API Function** action.
2. On the **Copy Function** page, enter or use the AssistButton in the **Target API Set Code** field to select the API set that will contain the new function.
3. In the **Target Function Code** field, enter a code to identify the new API function as unique.

4. On the **Data To Copy** FastTab, choose the types of data from the source API function that you want to copy to the new record.
5. Choose the **OK** button.

A new API function is created from the selected source function.


API Messages

All content related to an API call is entered as a separate “API message” in Business Central. Each API message contains an API call’s response and request data, as well as other information. This can assist in testing and troubleshooting activities, as it presents all information related to a given API call in a single place.

There are five stages to an API message’s “lifecycle:”

1. **Message Initialization:** the API function that will manage the API call is chosen, and any related API parameters are added.
2. **Build Request:** the request is built, based on the configuration of the related API function.
3. **Send Request:** the request is sent from Business Central to the external system/platform.
4. **Receive Response:** a response to the API call is retrieved into Business Central from the external system/platform.
5. **Process Response:** the API message’s [data buffer](#) is built from the response data, and data is processed in Business Central according to any related [mappings](#).

To view API messages:

1. Choose the  icon, enter **API Messages**, and then choose the related link.
2. Open the API message you want to review. Depending on the frequency of API function execution, there may be a large volume of API messages. Use filters and sorting to locate the necessary record.

You can also view all of the API messages that have generated for a specific API function from the **API Functions** page by selecting an API function line, choosing the **Home** ribbon and then choosing the **API Messages** action.

On the **API Message** page, different types of data related to the API message are presented on separate FastTabs:

- **General:** basic information about the API message, including the API set and API function for which it was created and various status values.
- **Processing Timestamps:** start and finish times for the execution of the related request, data buffer, and response processes.
- **Related API Messages:**
- **Parameters:** any parameters have been defined as part of the related API call. For more information on these parameter lines, click [here](#).
- **Request:** the data contained within the request that was sent from Business Central to the external system.

- **Response:** the data contained within the response that was retrieved into Business Central from the external system. The response data is presented in a “raw” character stream on this FastTab; you can view the message’s [data buffer](#) to see a more structured presentation of the message response.
- **Query Request:** the content of the query that was sent from Business Central to the external system.

In practice, an API is completely processed when it is created due to the execution of an API function. However, for testing purposes, it is possible to manually set up an API message. You can then execute this message directly from the **API Message** page by choosing the **Actions** ribbon, then choosing one of the following options:

- **Execute:** this will instruct the API Engine to execute the API message in its entirety. The API message will be fully processed.
- **Manual Processing:** this submenu presents different stages of the API message’s lifecycle. In this way, you can manually step through each stage to review the result.

Adding API Parameters to an API Message

An API message will oftentimes have one or more parameters influence the nature of the API call. Each parameters presented in a separate line on the **Parameters** FastTab.

In general, parameters are determined through the configuration that has taken place for the related API function and automatically entered on the **Parameters** FastTab. However, when developing or testing functionality, it may be useful to make parameter adjustments directly to an API message. After making these adjustments, you can execute the API message and review the results.

To add an API parameter to an API message:

1. On the **Parameters** FastTab, choose the **Add New Parameter** action.
2. Choose one of the available parameter options:
 - **Table Record:** the API Engine will present a list of tables in Business Central. Locate the desired table, then choose the value in the **ID** field to view a list of records in that table. From here, you can choose the record that you want to send as part of the API call’s request. Note that many API calls a case-sensitive; it is recommended that you choose a record’s description, rather than code, when adding a table parameter.
 - **Key Value:** enter the parameter key (the type of information you want to send as part of the API call’s request) and its value (the specific value of that information type you want to send) in the respective **Key** and **Value** fields.

Message Data Buffers

The content of an API call’s response from the external system is presented in two formats:

- A “raw” character stream is presented in the **Response** FastTab on the **API Message** page.

- A more structured version of the response data is entered as a “message data buffer.” This presents the data according to parent-child relationships, field designation, and so forth. Message data buffers can be processed via JSON or XML; this setting is defined when you are [setting up the related API function](#).

To view a message data buffer from the **API Message** page, choose the **Related** ribbon, then choose the **Message Data Buffer** action.

An API message’s data buffer presents the response in a tabular format, with separate lines being created for each element of the response. Every line has information about the element’s name and value; path, depth, and parent entry number (this can be used to identify parent-child relationships in the data); data type; and so forth.