Using Callable Pipelines to consolidate Create/Update Triggers

Introduction

With automations there is the ability to create a single trigger for both new and updated records. This is not available in Pipelines to ensure Pipelines will only fire when necessary. There are two options to achieve a similar outcome in Pipelines.

- 1. Create two separate Pipelines. One Pipeline with the Create trigger followed by the actions needed to accomplish the goal of the Pipeline, and a second Pipeline with the Update trigger with the same actions as the first Pipeline. For pipelines that require complicated and lengthy steps, this can be cumbersome to build, creates duplication of efforts, and pipelines can become difficult to manage. Because of this, it is not generally recommended this approach is taken.
- 2. Create three separate Pipelines. One pipeline with the Create trigger followed by the Call another pipeline action, and a Second pipeline with the Update trigger followed by the Call another pipeline action. The third pipeline will use the Pipeline Called Trigger, followed by the actions needed to accomplish the goal of the Pipeline. This method is better for pipelines that might require more complexity and many steps to complete, as the third pipeline will hold all the complexity, and becomes easier to manage long-term.

In this example, imagine you want to create a change history log table in Quick Base to track Project changes. You want to capture the status of a Project when it is created, but also capture the status when a Project is updated. One option would be to take the approach detailed in option #1 above. This is described in detail in the Creating data change logs with pipelines guide. However, for simplicity of this guide, imagine there was much more complexity needed in the Pipeline. This guide follows option #2.

Overview

Assume the following table structure:

Projects -> Project History Log

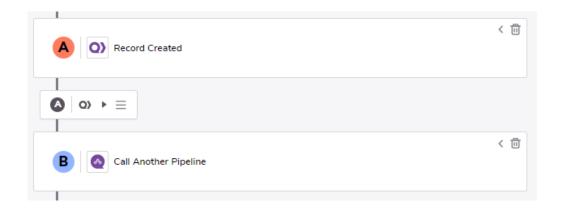
Scenario: If a Project is created or updated, capture the Project's status, date/time created/modified, and who created/modified the project.

Overview of example pipelines

The example requires three Pipelines. The first two are quite simple, while the third will hold the actions necessary to carry out the goal of the Pipeline.

The **first** pipeline will have the following steps:

- Step A: Trigger the pipeline when a record in the Projects table is created
- Step B: Call another Pipeline to carry out the actions



The **second** pipeline will have the following steps:

- Step A: Trigger the pipeline when a record in the Projects table is updated
- Step B: Call another Pipeline to carry out the actions



The **third** pipeline will have the following steps:

- Step A: Use the Pipeline Called trigger to trigger the pipeline that will carry at the actions of the pipeline
- Step B: Create Record in the Project History Log table

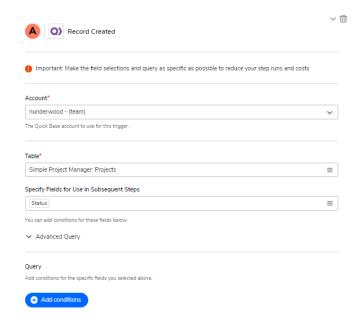


Note: With this method, the **third** pipeline would typically have much more complexity to carry out. For simplicity of this guide, a single action will be used.

Guided Instructions

Pipeline #1, Step A: Trigger when Project is Created

The goal of Step A is to trigger the pipeline whenever a Project is Created.



Setup:

- 1. Create a new pipeline and give it a descriptive name
- 2. Open the Quick Base channel, then open the Records category
- 3. Drag the **Record Created** trigger onto the canvas
- 4. In the Account box, select your user token or enter a new one (click here to learn how).
- 5. For this example, we'll select the **Projects** table, which is the parent record in our use-case, and the table whose status change we want to log
- 6. In the Specify Fields for Use in Subsequent Steps box, select the **Status** field. This will allow us to leverage the Status field's value later in the pipeline

 Note: If using a custom key field, that will need to be defined here

Pipeline #1, Step B: Call another Pipeline

The goal of Step B is to Call another Pipeline and pass the needed values from Step A to the pipeline being called.

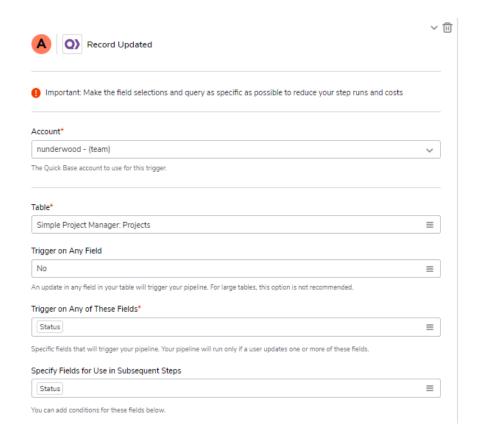
| B Call A | nother Pipeline |
|--|--|
| Call Definition* | |
| log_data_change(stat | us, <u>lastmodby</u> , <u>recordID</u>) |
| created_at) . Allowed sym letter. The definition should | following format <pre>call_name(arg1[, arg2, arg3])</pre> For example <pre>myFunction(contact_id, bols are latin alphanumeric characters and underscore. Call and argument names should start with a loe identical between caller and called. In other words you should <pre>copy/paste</pre> the definition between '* and *"Pipeline Called"* steps.</pre> |
| Status | |
| {{a.status}} | |
| Lastmodby | |
| {{a.last_modified_by.id | d}} |
| Record | |
| | |

Set up:

- 1. Open the **Callable pipelines** channel
- 2. Drag the **Call Another Pipeline** action onto the Pipeline Designer
- 3. In the Call Definition box, specify both a call name and field names
 - a. **Call name**: this must start with a letter. You can specify anything you want here. A brief description of the call will work, for example "log_data_change"
 - b. **Field names**: Begin with an open parenthesis following your Call name. The first character of each field name must start with a letter. Again, you can specify anything you want here. In our example, we want to pass the Status field value from our trigger to the pipeline being called, so it makes sense to type in "status". Each field name must be separated by a comma. We also want the Last Modified By and Record ID (or custom key field) of our trigger. After you've defined your field names, close the parenthesis
 - c. The Call Definition can look something like below: log data change(status, lastmodby, recordID)
 - d. Click outside the call definition box, the fields will appear as input fields below
 Note: The input fields that appear may differ slightly than the field names you defined in step 3b.
- 4. Populate the input field boxes with the appropriate data values from Step A by dragging the field values from the available fields panel.

Pipeline #2, Step A: Trigger when Project is Updated

The goal of Step A is to trigger the pipeline whenever a Project is Updated AND the Status field changes



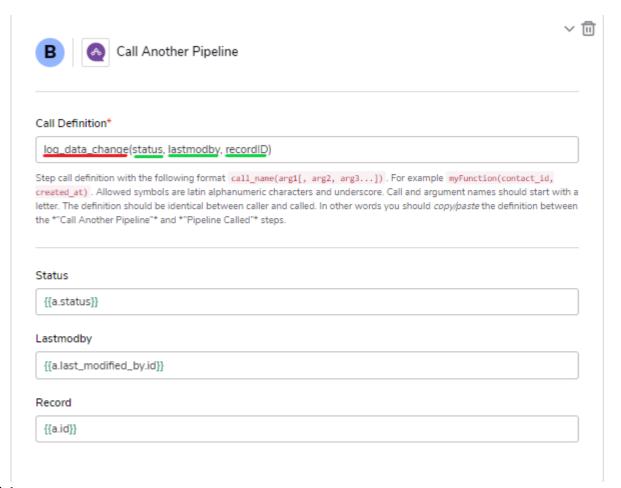
Set up:

- 1. Create a new pipeline and give it a descriptive name
- 2. Open the Quick Base channel, then open the Records category
- 3. Drag the **Record Updated** trigger onto the canvas
- 4. In the Account box, select your user token or enter a new one (click here to learn how).
- 5. For this example, we'll select the **Projects** table, which is the parent record in this use-case, and the table whose status change needs to be tracked
- 6. Set Trigger on Any Field to No
- 7. In the *Trigger on Any of These Fields* box, select **Status Note**: Steps 6 & 7 ensure this Pipeline only triggers when necessary. These steps should be taken whenever possible in pipeline creation for step-run efficiency
- 8. In the *Specify Fields for Use in Subsequent Steps* box, define the **Status** field. This will allow you to leverage the Status field's value later in the pipeline

Pipeline #2, Step B: Call another Pipeline

The goal of Step B is to Call another Pipeline and pass the needed values from Step A to the pipeline being called.

Note: this step will be the exact same at Step B in pipeline #1. Copying and Pasting the call definition from pipeline #1 into the call definition of pipeline #2 is strongly recommended. For this pipeline to run successfully, the call definitions must be the exact same.



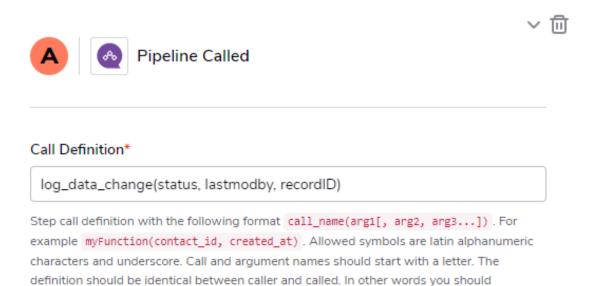
Set up:

- 1. Open the Callable pipelines channel
- 2. Drag the **Call Another Pipeline** action onto the Pipeline Designer
- 3. In the Call Definition box, specify both a call name and field names
 - **a.** Call name: this must start with a letter. You can specify anything you want here. A brief description of the call will work, for example "log_data_change"
 - b. Field names: Begin with an open parenthesis following your Call name. The first character of each field name must start with a letter. Again, you can specify anything you want here. In our example, we want to pass the Status field value from our trigger to the pipeline being called, so it makes sense to type in "status". Each field name must be separated by a comma. We also want the Last Modified By and Record ID (or custom key field) of our trigger. After you've defined your field names, close the parenthesis
 - c. The Call Definition can look something like below: log_data_change(status, lastmodby, recordID)
 - d. Click outside the call definition box, the fields will appear as input fields below **Note**: The input fields that appear may differ slightly than the field names you defined in step 3b.

4. Populate the input field boxes with the appropriate data values from Step A by dragging the field values from the available fields panel.

Pipeline #3, Step A: Trigger Call Another Pipeline

The goal of Step A is to trigger the Call Another Pipeline trigger



Set up:

steps.

- 1. Create a new pipeline and give it a descriptive name
- 2. Open the **Callable Pipelines** channel
- 3. Drag the **Pipeline Called** trigger onto the Pipeline Designer
- 4. In the Call Definition box, Copy and Paste the Call definition from Pipeline #1 or #2 (they should be the same)

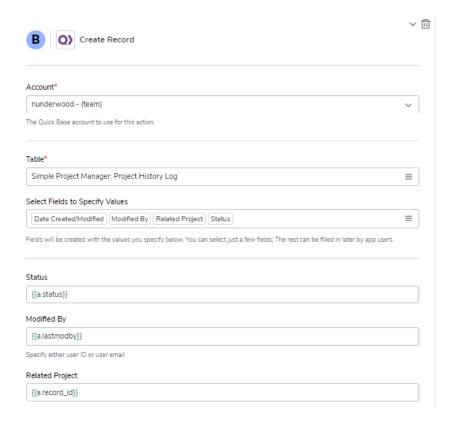
copy/paste the definition between the *"Call Another Pipeline"* and *"Pipeline Called"*

I. It is important that the call definitions match exactly for this pipeline to run successfully

Pipeline #3, Step B: Create Record

The goal of Step B is to Create a record in the Project History Log table

Note: For simplicity of this guide, in this example there is only one step here. This method is better utilized for Pipelines that have complex logic and many steps after the triggers.



Set up:

- 1. Drag the Create Record step onto the canvas as Step B
- 2. Select your user token again
- 3. Select the Project History Log table
- 4. In the Select Fields to Specify Values box, choose the field values that you want to populate
- 5. Drag field values from Step A into the corresponding input fields