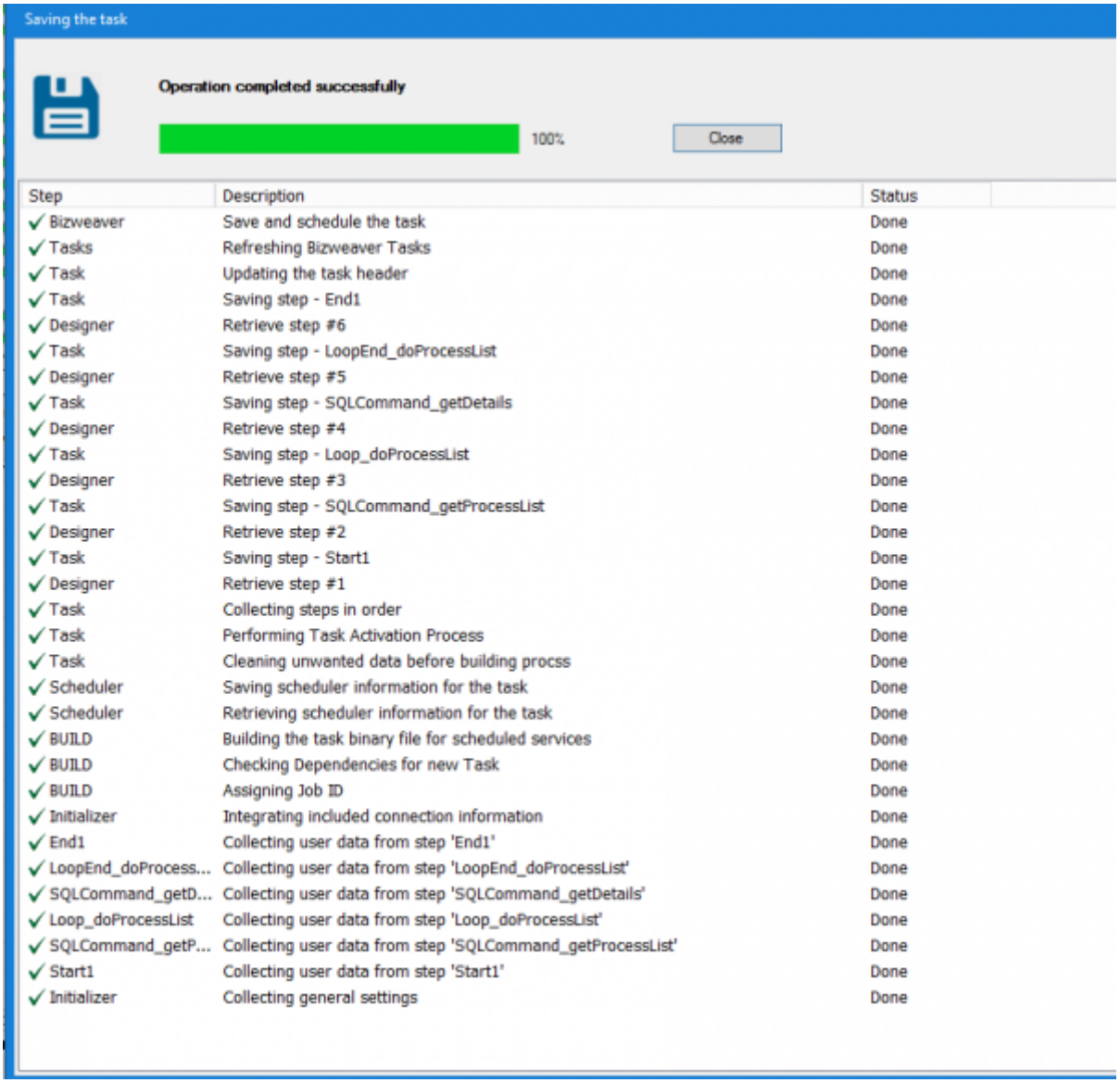
# Save a Bizweaver Workflow

To save a workflow, click the “Save” icon in the top icon bar.

New in Bizweaver version 2.4

## Successful Save

When a workflow is saved (or duplicated), a new window is displayed. Details about the save process are displayed as various validations, of each step take place. An example of a successful save is shown below. Note that all rows show a green checkbox on a successful save.



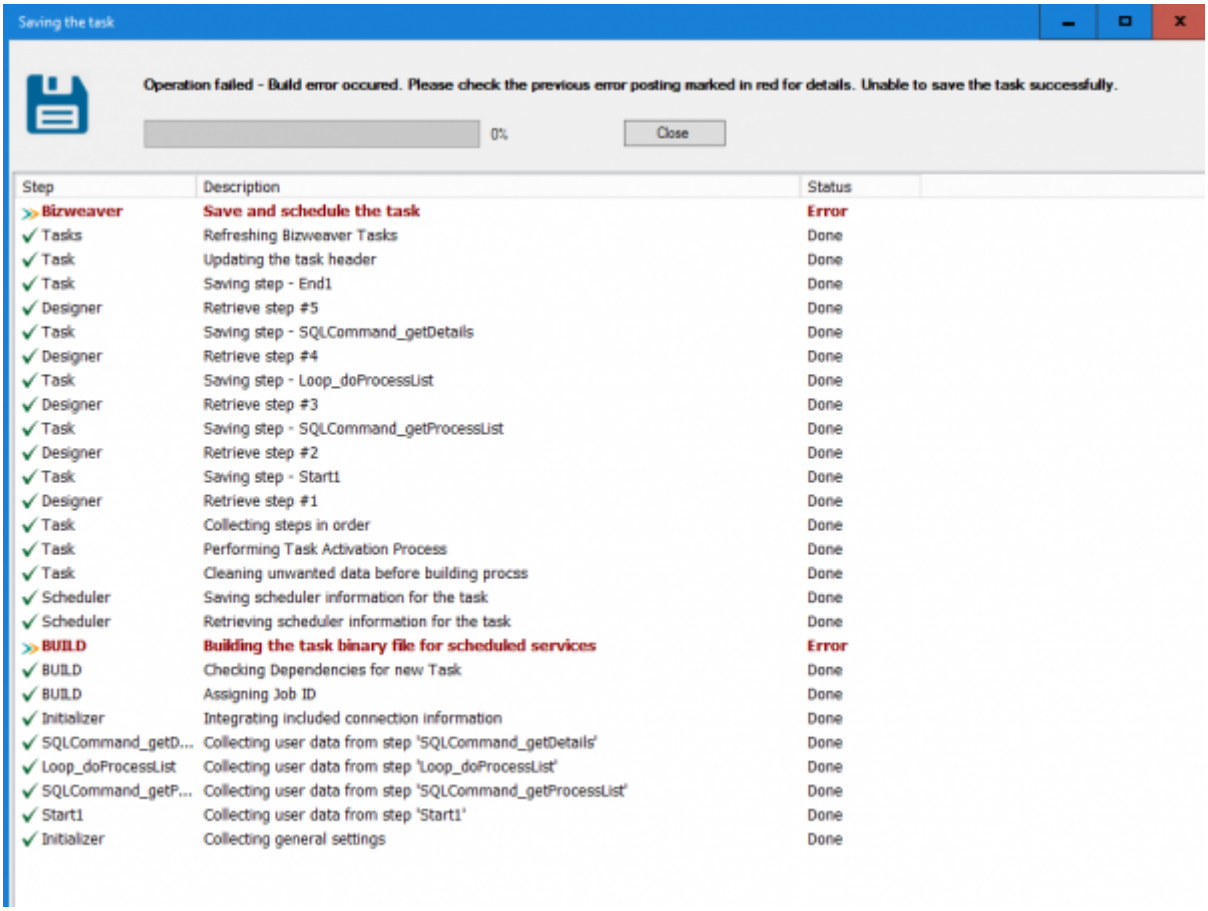

| Step | Description | Status |
|---|---|---|
| ✓ Bizweaver | Save and schedule the task | Done |
| ✓ Tasks | Refreshing Bizweaver Tasks | Done |
| ✓ Task | Updating the task header | Done |
| ✓ Task | Saving step - End1 | Done |
| ✓ Designer | Retrieve step #6 | Done |
| ✓ Task | Saving step - LoopEnd_doProcessList | Done |
| ✓ Designer | Retrieve step #5 | Done |
| ✓ Task | Saving step - SQLCommand_getDetails | Done |
| ✓ Designer | Retrieve step #4 | Done |
| ✓ Task | Saving step - Loop_doProcessList | Done |
| ✓ Designer | Retrieve step #3 | Done |
| ✓ Task | Saving step - SQLCommand_getProcessList | Done |
| ✓ Designer | Retrieve step #2 | Done |
| ✓ Task | Saving step - Start1 | Done |
| ✓ Designer | Retrieve step #1 | Done |
| ✓ Task | Collecting steps in order | Done |
| ✓ Task | Performing Task Activation Process | Done |
| ✓ Task | Cleaning unwanted data before building procss | Done |
| ✓ Scheduler | Saving scheduler information for the task | Done |
| ✓ Scheduler | Retrieving scheduler information for the task | Done |
| ✓ BUILD | Building the task binary file for scheduled services | Done |
| ✓ BUILD | Checking Dependencies for new Task | Done |
| ✓ BUILD | Assigning Job ID | Done |
| ✓ Initializer | Integrating included connection information | Done |
| ✓ End1 | Collecting user data from step 'End1' | Done |
| ✓ LoopEnd_doProcess... | Collecting user data from step 'LoopEnd_doProcessList' | Done |
| ✓ SQLCommand_getD... | Collecting user data from step 'SQLCommand_getDetails' | Done |
| ✓ Loop_doProcessList | Collecting user data from step 'Loop_doProcessList' | Done |
| ✓ SQLCommand_getP... | Collecting user data from step 'SQLCommand_getProcessList' | Done |
| ✓ Start1 | Collecting user data from step 'Start1' | Done |
| ✓ Initializer | Collecting general settings | Done |

# Error in Save

For the most part, the information displayed is not of interest to the user; except when an error is encountered. However, when an error is encountered the results will assist with identifying the issue. Following is an example of the error handling process.

This is the workflow being saved. In this case the LoopEnd step is missing.

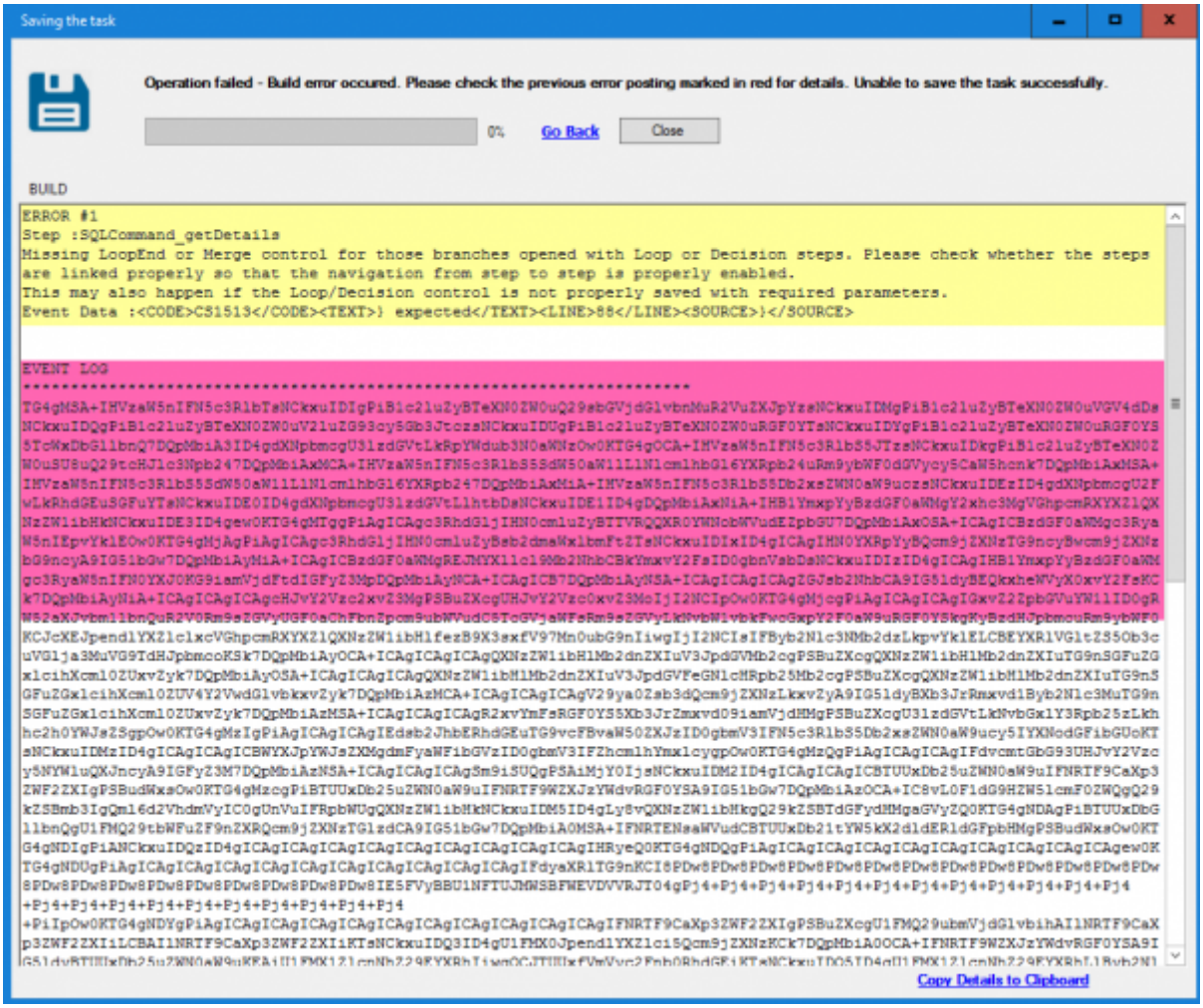

The response now indicates that error(s) were encountered. Points where an error was found are highlighted in red.




| Step | Description | Status |
|---|---|---|
| Bizweaver | Save and schedule the task | Error |
| Tasks | Refreshing Bizweaver Tasks | Done |
| Task | Updating the task header | Done |
| Task | Saving step - End1 | Done |
| Designer | Retrieve step #5 | Done |
| Task | Saving step - SQLCommand_getDetails | Done |
| Designer | Retrieve step #4 | Done |
| Task | Saving step - Loop_doProcessList | Done |
| Designer | Retrieve step #3 | Done |
| Task | Saving step - SQLCommand_getProcessList | Done |
| Designer | Retrieve step #2 | Done |
| Task | Saving step - Start1 | Done |
| Designer | Retrieve step #1 | Done |
| Task | Collecting steps in order | Done |
| Task | Performing Task Activation Process | Done |
| Task | Cleaning unwanted data before building procss | Done |
| Scheduler | Saving scheduler information for the task | Done |
| Scheduler | Retrieving scheduler information for the task | Done |
| BUILD | Building the task binary file for scheduled services | Error |
| BUILD | Checking Dependencies for new Task | Done |
| BUILD | Assigning Job ID | Done |
| Initializer | Integrating included connection information | Done |
| SQLCommand_getD... | Collecting user data from step 'SQLCommand_getDetails' | Done |
| Loop_doProcessList | Collecting user data from step 'Loop_doProcessList' | Done |
| SQLCommand_getP... | Collecting user data from step 'SQLCommand_getProcessList' | Done |
| Start1 | Collecting user data from step 'Start1' | Done |
| Initializer | Collecting general settings | Done |

The first error is typical very general in nature. The error points further down in the listing will help identify the issue.

Click on a error row to display the details. Note that the yellow and red sections in the image are for illustration only. The actual output will not display these colors.




A general description of the error is displayed in the area at the top of “Build” section (marked in yellow in the image). This will often times point you to the issue that needs to be fixed.

The remaining information shown (starting with the area marked in red in the image) is encrypted process information that can only be viewed by Third Wave support personnel. For support purposes you may be requested to use the Copy Details to Clipboard link at the bottom of the display area. This will copy the information to your clipboard where it can be copied to a file or the body of an email. Having this information will allow support personnel to see more detail about the issue.

# Duplicating a Workflow

Duplicating a workflow allows a user to take an active workflow and copy it to the same location or another location that is specified by the user. The workflow name will stay the same but the workflow ID will change.

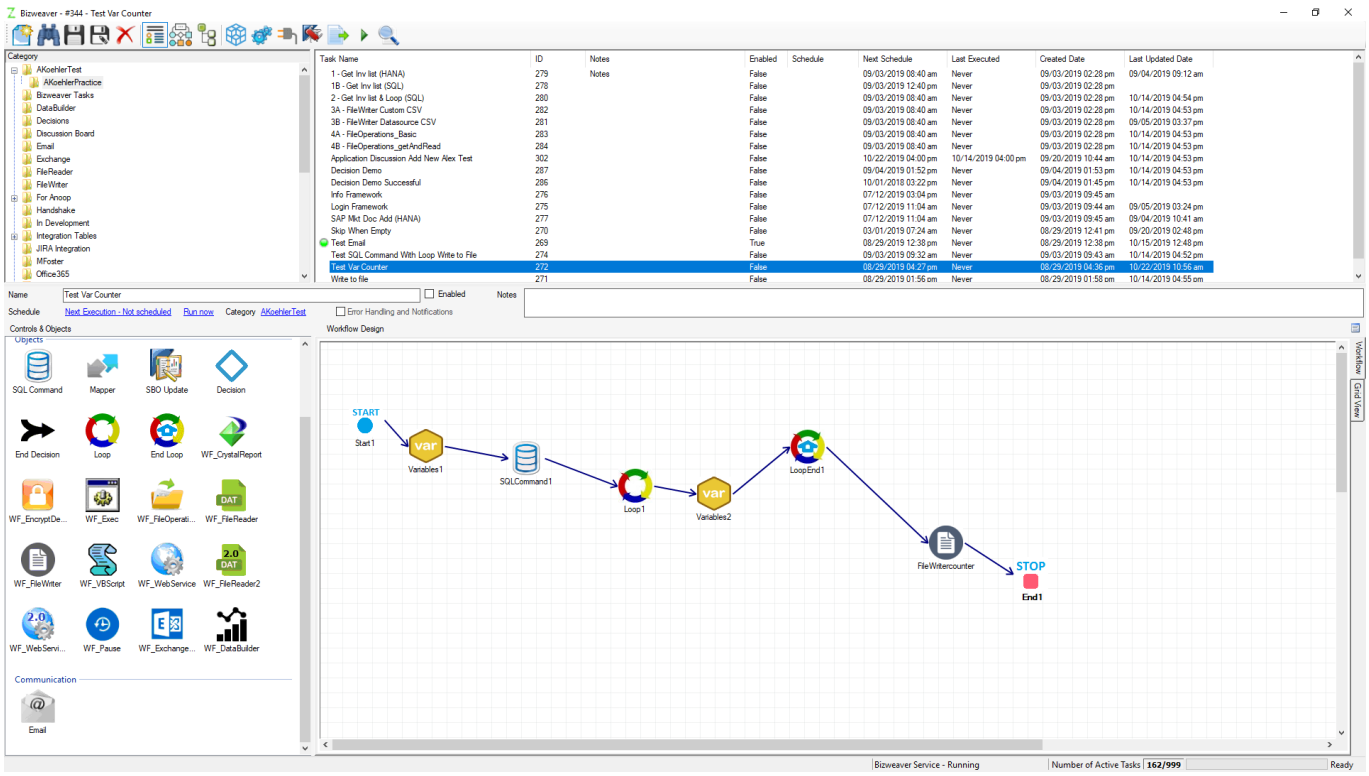
### Use Case

This functionality allows users to copy a workflow, so they can add new steps or modify it without effecting the initial workflow.

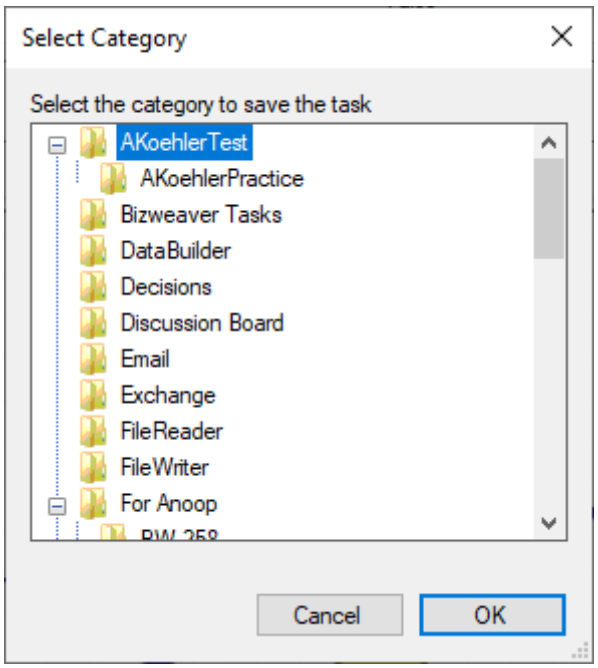
# Duplicating Workflow Example

Below are the steps to duplicate a workflow.

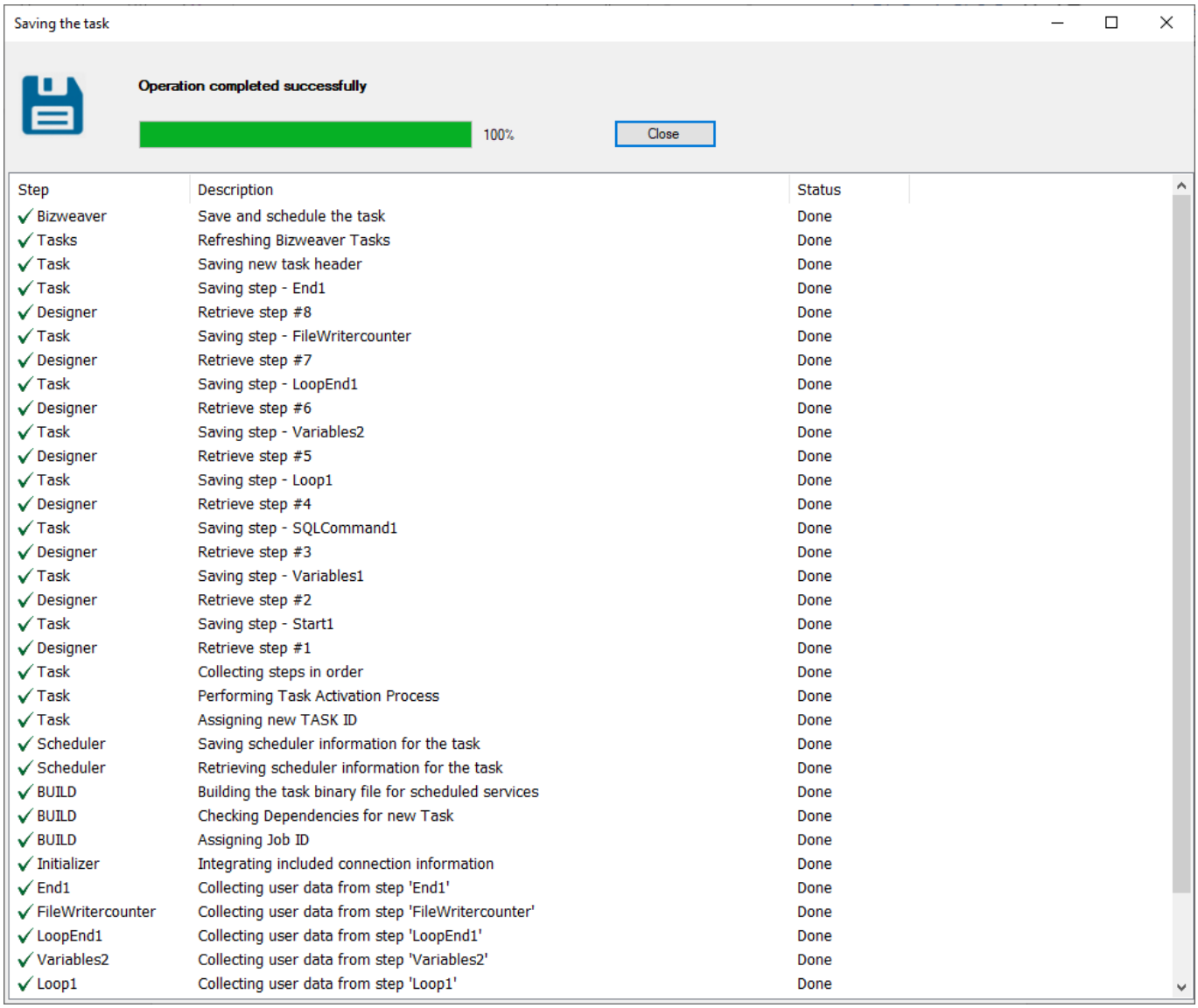
- Open up the workflow that you want to duplicate. Note we are in AKoehlerPractice folder in the below image. The workflow we are working with is Name: Test Var Counter, ID: 272




- Select the duplicate Button  (This will open a new window below)



- Select the folder you want to save it to. In this instance we will save it to the Root folder of AKoeherTest. Press “OK”
- The Workflow will bring up the save screen as it saves to the new location.




| Step | Description | Status |
|---|---|---|
| ✓ Bizweaver | Save and schedule the task | Done |
| ✓ Tasks | Refreshing Bizweaver Tasks | Done |
| ✓ Task | Saving new task header | Done |
| ✓ Task | Saving step - End1 | Done |
| ✓ Designer | Retrieve step #8 | Done |
| ✓ Task | Saving step - FileWritercounter | Done |
| ✓ Designer | Retrieve step #7 | Done |
| ✓ Task | Saving step - LoopEnd1 | Done |
| ✓ Designer | Retrieve step #6 | Done |
| ✓ Task | Saving step - Variables2 | Done |
| ✓ Designer | Retrieve step #5 | Done |
| ✓ Task | Saving step - Loop1 | Done |
| ✓ Designer | Retrieve step #4 | Done |
| ✓ Task | Saving step - SQLCommand1 | Done |
| ✓ Designer | Retrieve step #3 | Done |
| ✓ Task | Saving step - Variables1 | Done |
| ✓ Designer | Retrieve step #2 | Done |
| ✓ Task | Saving step - Start1 | Done |
| ✓ Designer | Retrieve step #1 | Done |
| ✓ Task | Collecting steps in order | Done |
| ✓ Task | Performing Task Activation Process | Done |
| ✓ Task | Assigning new TASK ID | Done |
| ✓ Scheduler | Saving scheduler information for the task | Done |
| ✓ Scheduler | Retrieving scheduler information for the task | Done |
| ✓ BUILD | Building the task binary file for scheduled services | Done |
| ✓ BUILD | Checking Dependencies for new Task | Done |
| ✓ BUILD | Assigning Job ID | Done |
| ✓ Initializer | Integrating included connection information | Done |
| ✓ End1 | Collecting user data from step 'End1' | Done |
| ✓ FileWritercounter | Collecting user data from step 'FileWritercounter' | Done |
| ✓ LoopEnd1 | Collecting user data from step 'LoopEnd1' | Done |
| ✓ Variables2 | Collecting user data from step 'Variables2' | Done |
| ✓ Loop1 | Collecting user data from step 'Loop1' | Done |

- Once saved, you can now navigate to the folder it was saved in. Note the name stayed the same but the ID changes from 272 to 344.