Understanding Pipeline Failures and Error Handling

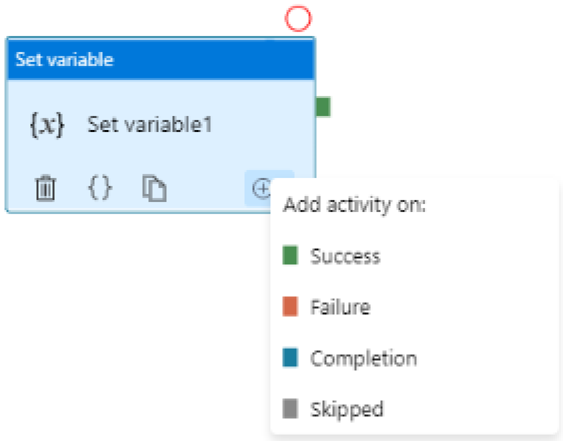
Last updated by | Jackie Huang | Jan 4, 2022 at 12:24 AM PST

Understanding Pipeline Failures and Error Handling

Author: chez

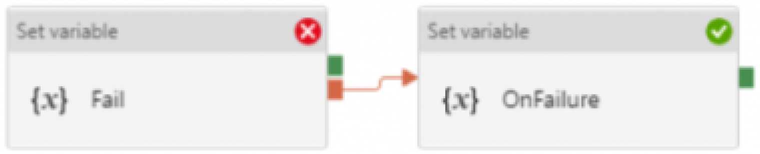
Article Link

Azure Data Factory orchestration allows conditional logic and enables user to take different based upon outcomes of a previous activity. In total we allows four conditional paths: Upon Success (default pass), Upon Failure, Upon Completion, and Upon Skip. Using different paths allow users to build robust pipelines and incorporates error handling in their ETL/ELT logic.

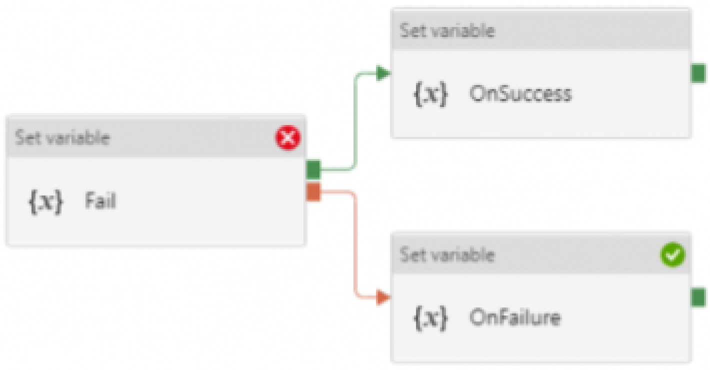


Here are two common error handling pattern we see customers use:

1. **TRY-CATCH block.** Define the business logic, and only defines *Upon Failure* path to catch any error from previous activities



2. **DO-IF-ELSE block.** Define the business logic, and depends on the outcome of the activity, enacts either *Upon Success* path or *Upon Failure* path



Both are valid ways to incorporate error handling into the pipeline. However, upon pipeline execution, they may show different outcomes.

Approach #1 , TRY-CATCH, shows pipeline **succeeds** if *Upon Failure* path clears, where as approach #2 , DO-IF-ELSE show pipeline **failed** if *Upon Failure* path is enacted.

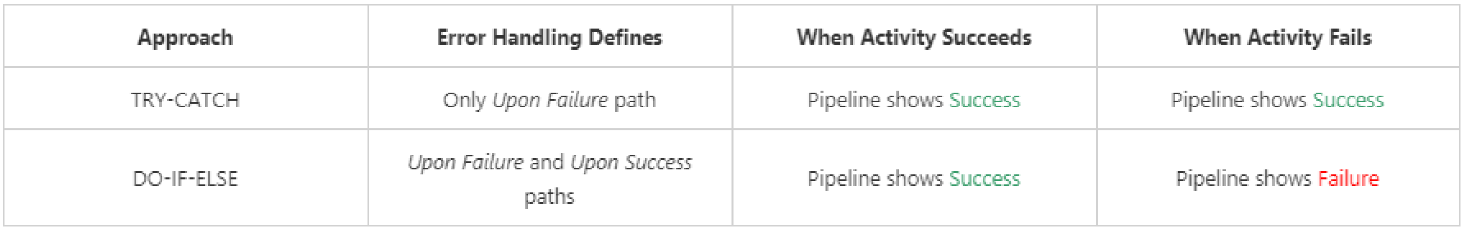
Technical reasons for the difference is that, Azure Data Factory defines pipeline success and failures as follows:

- Evaluate outcome for all leaves activities. If a leaf activity was skipped, we evaluate its parent activity instead
- Pipeline result is success if and only if all leaves succeed

Applying the logic to previous examples.

1. In approach #1 TRY-CATCH block:
    - when previous activity succeeds: the node activity, *Upon Failure*, is skipped and its parent node succeeds, so overall pipeline succeeds
    - when previous activity fails: the node activity, *Upon Failure*, enacted and overall pipeline succeeds if *Upon Failure* path succeeds
2. In approach #2 DO-IF-ELSE block:
    - when previous activity succeeds: one node activity, *Upon Success*, succeeded, and the other node activity, *Upon Failure*, is skipped and its parent node succeeds; so overall pipeline succeeds
    - when previous activity fails: one node activity, *Upon Success*, is skipped and its parent node failed; so overall pipeline *failed*

Here is a table summarizing the difference



Approach	Error Handling Defines	When Activity Succeeds	When Activity Fails
TRY-CATCH	Only *Upon Failure* path	Pipeline shows Success	Pipeline shows Success
DO-IF-ELSE	*Upon Failure* and *Upon Success* paths	Pipeline shows Success	Pipeline shows Failure