SSIS 2012, Part 08 of 11: Handling Errors and Events

Meet the expert: Don Kiely is a featured instructor on many of our SQL Server and Visual Studio courses. He is a nationally recognized author, instructor, and consultant specializing in Microsoft technologies. Don has many years of teaching experience, is the author or co-author of several programming books, and has spoken at many industry conferences and user groups. In addition, Don is a consultant for a variety of companies that develop distributed applications for public and private organizations.

Prerequisites: This course assumes that you have a basic familiarity with the concept of relational databases and a basic understanding of what SQL Server is and the high-level tools in it, as well as how to create and manage objects using Management Studio. You should also have a basic understanding of how SQL Server implements security, including its authentication and authorization schemes, and how to assign permissions on securable objects to principals. You should know the fundamentals of Transact-SQL to write queries to retrieve data and join data from multiple tables, and how to execute scripts using the query editor in Management Studio. You must also know how to connect to an instance of SQL Server 2012 or 2014 using the various connection dialog boxes in Management Studio and development tools. It will be very helpful, but not absolutely necessary, to have experience with .NET development using Visual Studio 2013 or later for the portions of the course that deal with SQL Server Data Tools (which is a lot of it). At the very least, we’ll assume that you are well familiar with the Visual Studio user interface. This course assumes no prior knowledge of SQL Server Integration Services beyond what is covered in the previous SSIS courses.

Runtime: 02:05:07

Course description: Integration Services has a variety of tools and provides various techniques you can use to see what a task or package is doing and how it is doing it, as you’ll see in this course. You can cause the behavior or a task to change when problems arise, whether from bad data or missing resources, and you can partition a Control Flow so that a whole separate set of tasks execute only when certain kinds of events occur. You can pause package execution at pre-defined breakpoints—conceptually similar to breakpoints in programming tools but quite different in practice. You cause the package to store information in a checkpoint file about where it failed so that the next time you run the package it doesn’t repeat any tasks that completed successfully. And you can log many kinds of information about the package as each task executes. In this course, you’ll learn about these debugging, error handling, and logging features that can help you diagnose most kinds of problems in your package.

Course outline:

Execution With Checkpoints
- Introduction
- When Things Go Wrong
- Restarting Package Execution
- Implementing Checkpoints
- How it Works
- Demo: Using Checkpoints
- Demo: Creating a Problem
- Demo: Execution Scenario
- Demo: The Checkpoint File
- Demo: Restart Package Execution
- Summary

Error Handling and Debugging
- Introduction
- Precedence Constraints
- Demo: Introducing Random Errors
- Demo: Adding a Failure Path
- Demo: Two Failure Paths
- Demo: The Constraint Editor
- Summary

Error Outputs In Data Flows
- Introduction
- Error Actions
- Demo: Data Conversion Pkg
- Demo: Data Conversion Errors
- Demo: Configure Error Output
- Demo: Redirect Bad Data
- Summary

Breakpoints
- Introduction
- Demo: For Loop Container
- Demo: Setting Breakpoints
- Summary

Logging Events
- Introduction
- Demo: The Data Base & Pkg
- Demo: Log Events
- Summary

Configuring Package Logging
- Introduction
- Configuring Package Logging
- Logging to a Text File
- Demo: Configure Logging
- Demo: Execute Pkg
- Summary

Configure for Windows Event Log
- Introduction
- Demo: Windows Event Log
- Custom Logging
- Summary

Event Handling
- Introduction
- Event Handling
- Handling Errors
- Demo: Event Handling Project
- Demo: For Loop Container
- Demo: Execute the Package
- Demo: Create an Event Handler
- Demo: Execute & Note Results
- Summary