SSIS How to Create an ETL Package

10/1/2018 • 3 minutes to read • Edit Online

> For content related to previous versions of SQL Server, see SSIS Tutorial: Creating a Simple ETL Package.

In this tutorial, you learn how to use SSIS Designer to create a simple Microsoft SQL Server Integration Services package. The package that you create takes data from a flat file, reformats the data, and then inserts the reformatted data into a fact table. In following lessons, the package is expanded to demonstrate looping, package configurations, logging, and error flow.

When you install the sample data that the tutorial uses, you also install the completed versions of the packages that you create in each lesson of the tutorial. By using the completed packages, you can skip ahead and begin the tutorial at a later lesson if you like. If this tutorial is your first time working with packages or the new development environment, we recommend that you begin with Lesson1.

What is SQL Server Integration Services (SSIS)?

Microsoft SQL Server Integration Services (SSIS) is a platform for building high-performance data integration solutions, including extraction, transformation, and load (ETL) packages for data warehousing. SSIS includes graphical tools and wizards for building and debugging packages; tasks for performing workflow functions such as FTP operations, executing SQL statements, and sending e-mail messages; data sources and destinations for extracting and loading data; transformations for cleaning, aggregating, merging, and copying data; a management database, `SSISDB`, for administering package execution and storage; and application programming interfaces (APIs) for programming the Integration Services object model.

What You Learn

The best way to become acquainted with the new tools, controls, and features available in Microsoft SQL Server Integration Services is to use them. This tutorial walks you through SSIS Designer to create a simple ETL package that includes looping, configurations, error flow logic, and logging.

Prerequisites

This tutorial is intended for users familiar with fundamental database operations, but who have limited exposure to the new features available in SQL Server Integration Services.

To run this tutorial, you have to have the following components installed:

- SQL Server and Integration Services. To install SQL Server and SSIS, see Install Integration Services.
- The **AdventureWorksDW2012** sample database. To download the **AdventureWorksDW2012** database, download `AdventureWorksDW2012.bak` from AdventureWorks sample databases and restore the backup.
- The **sample data** files. The sample data is included with the SSIS lesson packages. To download the sample data and the lesson packages as a Zip file, see SQL Server Integration Services Tutorial Files.
  - Most of the files in the Zip file are read-only to prevent unintended changes. To write output to a file or to change it, you may have to turn off the read-only attribute in the file properties.
  - The sample packages assume that the data files are located in the folder `C:\Program Files\Microsoft SQL Server\100\Samples\Integration Services\Tutorial\Creating a Simple ETL Package`. If you unzip the download to another location, you may have to update the file path in multiple places

in the sample packages.

Lessons in This Tutorial

Lesson 1: Create a Project and Basic Package with SSIS
In this lesson, you create a simple ETL package that extracts data from a single flat file, transforms the data using lookup transformations and finally loads the result into a fact table destination.

Lesson 2: Adding Looping with SSIS
In this lesson, you expand the package you created in Lesson 1 to take advantage of new looping features to extract multiple flat files into a single data flow process.

Lesson 3: Add Logging with SSIS
In this lesson, you expand the package you created in Lesson 2 to take advantage of new logging features.

Lesson 4: Add Error Flow Redirection with SSIS
In this lesson, you expand the package you created in lesson 3 to take advantage of new error output configurations.

Lesson 5: Add SSIS Package Configurations for the Package Deployment Model
In this lesson, you expand the package you created in Lesson 4 to take advantage of new package configuration options.

Lesson 6: Using Parameters with the Project Deployment Model in SSIS
In this lesson, you expand the package you created in Lesson 5 to take advantage of using new parameters with the project deployment model.

Lesson 1: Create a Project and Basic Package with SSIS

10/1/2018 • 3 minutes to read • Edit Online

For content related to previous versions of SQL Server, see Lesson 1: Creating the Project and Basic Package.

In this lesson, you will create a simple ETL package that extracts data from a single flat file source, transforms the data using two lookup transformation components, and writes that data to the **FactCurrency** fact table in **AdventureWorksDW2012**. As part of this lesson, you will learn how to create new packages, add and configure data source and destination connections, and work with new control flow and data flow components.

IMPORTANT

This tutorial requires the **AdventureWorksDW2012** sample database. For more information on installing and deploying **AdventureWorksDW2012**, see Reporting Services Product Samples on CodePlex.

Understanding the Package Requirements

This tutorial requires Microsoft SQL Server Data Tools.

For more information on installing the SQL Server Data Tools see SQL Server Data Tools Download.

Before creating a package, you need a good understanding of the formatting used in both the source data and the destination. Once you understand both of these data formats, you will be ready to define the transformations necessary to map the source data to the destination.

Looking at the Source

For this tutorial, the source data is a set of historical currency data contained in the flat file, SampleCurrencyData.txt. The source data has the following four columns: the average rate of the currency, a currency key, a date key, and the end-of-day rate.

Here is an example of the source data contained in the SampleCurrencyData.txt file:

```
1.00070049USD9/3/05 0:001.001201442
1.00020004USD9/4/05 0:001
1.00020004USD9/5/05 0:001.001201442
1.00020004USD9/6/05 0:001
1.00020004USD9/7/05 0:001.00070049
1.00070049USD9/8/05 0:000.99980004
1.00070049USD9/9/05 0:001.001502253
1.00070049USD9/10/05 0:000.99990001
1.00020004USD9/11/05 0:001.001101211
1.00020004USD9/12/05 0:000.99970009
```

When working with flat file source data, it is important to understand how the Flat File connection manager interprets the flat file data. If the flat file source is Unicode, the Flat File connection manager defines all columns as [DT_WSTR] with a default column width of 50. If the flat file source is ANSI-encoded, the columns are defined as [DT_STR] with a column width of 50. You will probably have to change these defaults to make the string column types more appropriate for your data. To do this, you will need to look at the data type of the destination where the data will be written to and then choose the correct type within the Flat File connection manager.

Looking at the Destination

The ultimate destination for the source data is the **FactCurrency** fact table in **AdventureWorksDW**. The **FactCurrency** fact table has four columns, and has relationships to two dimension tables, as shown in the following table.

COLUMN NAME	DATA TYPE	LOOKUP TABLE	LOOKUP COLUMN
AverageRate	float	None	None
CurrencyKey	int (FK)	DimCurrency	CurrencyKey (PK)
DateKey	Int (FK)	DimDate	DateKey (PK)
EndOfDayRate	float	None	None

Mapping Source Data to be Compatible with the Destination

Analysis of the source and destination data formats indicates that lookups will be necessary for the **CurrencyKey** and **DateKey** values. The transformations that will perform these lookups will obtain the **CurrencyKey** and **DateKey** values by using the alternate keys from **DimCurrency** and **DimDate** dimension tables.

FLAT FILE COLUMN	TABLE NAME	COLUMN NAME	DATA TYPE
0	FactCurrency	AverageRate	float
1	DimCurrency	CurrencyAlternateKey	nchar (3)
2	DimDate	FullDateAlternateKey	date
3	FactCurrency	EndOfDayRate	float

Lesson Tasks

This lesson contains the following tasks:

- Step 1: Creating a New Integration Services Project
- Step 2: Adding and Configuring a Flat File Connection Manager
- Step 3: Adding and Configuring an OLE DB Connection Manager
- Step 4: Adding a Data Flow Task to the Package
- Step 5: Adding and Configuring the Flat File Source
- Step 6: Adding and Configuring the Lookup Transformations
- Step 7: Adding and Configuring the OLE DB Destination
- Step 8: Making the Lesson 1 Package Easier to Understand
- Step 9: Testing the Lesson 1 Tutorial Package

Start the Lesson

Step 1: Creating a New Integration Services Project

Lesson 1-1 - Creating a New Integration Services Project

10/1/2018 • 2 minutes to read • Edit Online

The first step in creating a package in Integration Services is to create an Integration Services project. This project includes the templates for the objects — data sources, data source views, and packages — that you use in a data transformation solution.

The packages that you will create in this Integration Services tutorial interpret the values of locale-sensitive data. If your computer is not configured to use the regional option English (United States), you need to set additional properties in the package. The packages that you use in lessons 2 through 5 are copied from the package created in lesson 1, and you need not update locale-sensitive properties in the copied packages.

> **NOTE**
>
> This tutorial requires Microsoft SQL Server Data Tools.
>
> For more information on installing the SQL Server Data Tools see SQL Server Data Tools Download.

To create a new Integration Services project

1. On the **Start** menu, point to **All Programs**, point to **Microsoft SQL Server**, and click **SQL Server Data Tools**.
2. On the **File** menu, point to **New**, and click **Project** to create a new Integration Services project.
3. In the **New Project** dialog box, expand the **Business Intelligence** node under **Installed Templates**, and select **Integration Services Project** in the **Templates** pane.
4. In the **Name** box, change the default name to **SSIS Tutorial**. Optionally, clear the **Create directory for solution** check box.
5. Accept the default location, or click **Browse** to browse to locate the folder you want to use. In the **Project Location** dialog box, click the folder and click **Select Folder**.
6. Click **OK**.

   By default, an empty package, titled **Package.dtsx**, will be created and added to your project under SSIS Packages.
7. In **Solution Explorer** toolbar, right-click **Package.dtsx**, click **Rename**, and rename the default package to **Lesson 1.dtsx**.

Next Task in Lesson

Step 2: Adding and Configuring a Flat File Connection Manager

Lesson 1-2 - Adding and Configuring a Flat File Connection Manager

10/1/2018 • 4 minutes to read • Edit Online

In this task, you add a Flat File connection manager to the package that you just created. A Flat File connection manager enables a package to extract data from a flat file. Using the Flat File connection manager, you can specify the file name and location, the locale and code page, and the file format, including column delimiters, to apply when the package extracts data from the flat file. In addition, you can manually specify the data type for the individual columns, or use the **Suggest Column Types** dialog box to automatically map the columns of extracted data to Integration Services data types.

You must create a new Flat File connection manager for each file format that you work with. Because this tutorial extracts data from multiple flat files that have exactly the same data format, you will need to add and configure only one Flat File connection manager for your package.

For this tutorial, you will configure the following properties in your Flat File connection manager:

- **Column names:** Because the flat file does not have column names, the Flat File connection manager creates default column names. These default names are not useful for identifying what each column represents. To make these default names more useful, you need to change the default names to names that match the fact table into which the flat file data is to be loaded.
- **Data mappings:** The data type mappings that you specify for the Flat File connection manager will be used by all flat file data source components that reference the connection manager. You can either manually map the data types by using the Flat File connection manager, or you can use the **Suggest Column Types** dialog box. In this tutorial, you will view the mappings suggested in the **Suggest Column Types** dialog box and then manually make the necessary mappings in the **Flat File Connection Manager Editor** dialog box.

The Flat File connection manager provides locale information about the data file. If your computer is not configured to use the regional option English (United States), you must set additional properties in the **Flat File Connection Manager Editor** dialog box.

To add a Flat File connection manager to the SSIS package

1. Right-click anywhere in the **Connection Managers** area, and then click **New Flat File Connection**.
2. In the **Flat File Connection Manager Editor** dialog box, for **Connection manager name**, type **Sample Flat File Source Data**.
3. Click **Browse**.
4. In the **Open** dialog box, locate the SampleCurrencyData.txt file on your machine.

   The sample data is included with the SSIS lesson packages. To download the sample data and the lesson packages, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Clear the Column names in the first data row checkbox.

To set locale sensitive properties

1. In the **Flat File Connection Manager Editor** dialog box, click **General**.
2. Set **Locale** to English (United States) and **Code page** to 1252.

To rename columns in the Flat File connection manager

1. In the **Flat File Connection Manager Editor** dialog box, click **Advanced**.
2. In the property pane, make the following changes:
   - Change the **Column 0** name property to **AverageRate**.
   - Change the **Column 1** name property to **CurrencyID**.
   - Change the **Column 2** name property to **CurrencyDate**.
   - Change the **Column 3** name property to **EndOfDayRate**.

> **NOTE**
> By default, all four of the columns are initially set to a string data type [DT_STR] with an **OutputColumnWidth** of 50.

To remap column data types

1. In the **Flat File Connection Manager Editor** dialog box, click **Suggest Types**.

   Integration Services automatically suggests the most appropriate data types based on the first 200 rows of data. You can also change these suggestion options to sample more or less data, to specify the default data type for integer or Boolean data, or to add spaces as padding to string columns.

   For now, make no changes to the options in the **Suggest Column Types** dialog box, and click **OK** to have Integration Services suggest data types for columns. This returns you to the **Advanced** pane of the **Flat File Connection Manager Editor** dialog box, where you can view the column data types suggested by Integration Services. (If you click **Cancel**, no suggestions are made to column metadata and the default string (DT_STR) data type is used.)

   In this tutorial, Integration Services suggests the data types shown in the second column of the following table for the data from the SampleCurrencyData.txt file. However, the data types that are required for the columns in the destination, which will be defined in a later step, are shown in the last column of the following table.

FLAT FILE COLUMN	SUGGESTED TYPE	DESTINATION COLUMN	DESTINATION TYPE
AverageRate	float [DT_R4]	FactCurrency.AverageRate	float
CurrencyID	string [DT_STR]	DimCurrency.CurrencyAlternateKey	nchar(3)
CurrencyDate	date [DT_DATE]	DimDate.FullDateAlternateKey	date
EndOfDayRate	float [DT_R4]	FactCurrency.EndOfDayRate	float

   The data type suggested for the **CurrencyID** column is incompatible with the data type of the field in the destination table. Because the data type of `DimCurrency.CurrencyAlternateKey` is nchar (3), **CurrencyID** must be changed from string [DT_STR] to Unicode string [DT_WSTR]. Additionally, the field `DimDate.FullDateAlternateKey` is defined as a date data type; therefore, **CurrencyDate** needs to be changed

from date [DT_Date] to database date [DT_DBDATE].

2. In the list, select the CurrencyID column and in the property pane, change the Data Type of column **CurrencyID** from string [DT_STR] to Unicode string [DT_WSTR].
3. In the property pane, change the data type of column **CurrencyDate** from date [DT_DATE] to database date [DT_DBDATE].
4. Click **OK**.

Next Task in Lesson

Step 3: Adding and Configuring an OLE DB Connection Manager

See Also

Flat File Connection Manager
Integration Services Data Types

Lesson 1-3 - Adding and Configuring an OLE DB Connection Manager

10/1/2018 • 2 minutes to read • Edit Online

After you have added a Flat File connection manager to connect to the data source, the next task is to add an OLE DB connection manager to connect to the destination. An OLE DB connection manager enables a package to extract data from or load data into any OLE DB–compliant data source. Using the OLE DB Connection manager, you can specify the server, the authentication method, and the default database for the connection.

In this lesson, you will create an OLE DB connection manager that uses Windows Authentication to connect to the local instance of **AdventureWorksDB2012**. The OLE DB connection manager that you create will also be referenced by other components that you will create later in this tutorial, such as the Lookup transformation and the OLE DB destination.

Add and configure an OLE DB Connection Manager to the SSIS package

1. Right-click anywhere in the **Connection Managers** area, and then click **New OLE DB Connection**.
2. In the **Configure OLE DB Connection Manager** dialog box, click **New**.
3. For **Server name**, enter **localhost**.

   When you specify localhost as the server name, the connection manager connects to the default instance of SQL Server on the local computer. To use a remote instance of SQL Server, replace localhost with the name of the server to which you want to connect.
4. In the **Log on to the server** group, verify that **Use Windows Authentication** is selected.
5. In the **Connect to a database** group, in the **Select or enter a database name** box, type or select **AdventureWorksDW2012**.
6. Click **Test Connection** to verify that the connection settings you have specified are valid.
7. Click **OK**.
8. Click **OK**.
9. In the **Data Connections** pane of the **Configure OLE DB Connection Manager** dialog box, verify that **localhost.AdventureWorksDW2012** is selected.
10. Click **OK**.

Next Task in Lesson

Step 4: Adding a Data Flow Task to the Package

See Also

OLE DB Connection Manager

Lesson 1-4 - Adding a Data Flow Task to the Package

10/1/2018 • 2 minutes to read • Edit Online

After you have created the connection managers for the source and destination data, the next task is to add a Data Flow task to your package. The Data Flow task encapsulates the data flow engine that moves data between sources and destinations, and provides the functionality for transforming, cleaning, and modifying data as it is moved. The Data Flow task is where most of the work of an extract, transform, and load (ETL) process occurs.

> **NOTE**
> SQL Server Integration Services separates data flow from control flow.

To add a Data Flow task

1. Click the **Control Flow** tab.
2. In the **SSIS Toolbox**, expand **Favorites**, and drag a **Data Flow Task** onto the design surfaceof the **Control Flow** tab.

   > **NOTE**
   > If the SSIS Toolbox isn't available, on the main menu select SSIS then SSIS Toolbox to display the SSIS Toolbox.

3. On the **Control Flow** design surface, right-click the newly added **Data Flow Task**, click **Rename**, and change the name to **Extract Sample Currency Data**.

   It is good practice to provide unique names to all components that you add to a design surface. For ease of use and maintainability, the names should describe the function that each component performs. Following these naming guidelines allows your Integration Services packages to be self-documenting. Another way to document your packages is by using annotations. For more information about annotations, see Use Annotations in Packages.

4. Right-click the Data Flow task, click **Properties**, and in the Properties window, verify that the **LocaleID** property is set to **English (United States)**.

Next Task in Lesson

Step 5: Adding and Configuring the Flat File Source

See Also

Data Flow Task

Lesson 1-5 - Adding and Configuring the Flat File Source

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will add and configure a Flat File source to your package. A Flat File source is a data flow component that uses metadata defined by a Flat File connection manager to specify the format and structure of the data to be extracted from the flat file by a transform process. The Flat File source can be configured to extract data from a single flat file by using the file format definition provided by the Flat File connection manager.

For this tutorial, you will configure the Flat File source to use the **Sample Flat File Source Data** connection manager that you previously created.

To add a Flat File Source component

1. Open the **Data Flow** designer, either by double-clicking the **Extract Sample Currency Data** data flow task or by clicking the **Data Flow tab**.
2. In the **SSIS Toolbox**, expand **OtherSources**, and then drag a **Flat File Source** onto the design surface of the **Data Flow** tab.
3. On the **Data Flow** design surface, right-click the newly added **Flat File Source**, click **Rename**, and change the name to **Extract Sample Currency Data**.
4. Double-click the Flat File source to open the Flat File Source Editor dialog box.
5. In the **Flat file connection manager** box, select **Sample Flat File Source Data**.
6. Click **Columns** and verify that the names of the columns are correct.
7. Click **OK**.
8. Right-click the Flat File source and click **Properties**.
9. In the Properties window, verify that the **LocaleID** property is set to **English (United States)**.

Next Task in Lesson

Step 6: Adding and Configuring the Lookup Transformations

See Also

Flat File Source
Flat File Connection Manager Editor (General Page)

Lesson 1-6 - Adding and Configuring the Lookup Transformations

10/1/2018 • 3 minutes to read • Edit Online

After you have configured the Flat File source to extract data from the source file, the next task is to define the Lookup transformations needed to obtain the values for the **CurrencyKey** and **DateKey**. A Lookup transformation performs a lookup by joining data in the specified input column to a column in a reference dataset. The reference dataset can be an existing table or view, a new table, or the result of an SQL statement. In this tutorial, the Lookup transformation uses an OLE DB connection manager to connect to the database that contains the data that is the source of the reference dataset.

> **NOTE**
> You can also configure the Lookup transformation to connect to a cache that contains the reference dataset. For more information, see Lookup Transformation.

For this tutorial, you will add and configure the following two Lookup transformation components to the package:

- One transformation to perform a lookup of values from the **CurrencyKey** column of the **DimCurrency** dimension table based on matching **CurrencyID** column values from the flat file.
- One transformation to perform a lookup of values from the **DateKey** column of the **DimDate** dimension table based on matching **CurrencyDate** column values from the flat file.

In both cases, the Lookup transformation will utilize the OLE DB connection manager that you previously created.

To add and configure the Lookup Currency Key transformation

1. In the **SSIS Toolbox**, expand **Common**, and then drag **Lookup** onto the design surface of the **Data Flow** tab. Place Lookup directly below the **Extract Sample Currency Data** source.
2. Click the **Extract Sample Currency Data** flat file source and drag the blue arrow onto the newly added **Lookup** transformation to connect the two components.
3. On the **Data Flow** design surface, click **Lookup** in the **Lookup** transformation, and change the name to **Lookup Currency Key**.
4. Double-click the **Lookup CurrencyKey** transformation to display the Lookup Transformation Editor.
5. On the **General** page, make the following selections:
   a. Select **Full cache**.
   b. In the **Connection type** area, select **OLE DB connection manager**.
6. On the **Connection** page, make the following selections:
   a. In the **OLE DB connection manager** dialog box, ensure that **localhost.AdventureWorksDW2012** is displayed.
   b. Select **Use results of an SQL query**, and then type or copy the following SQL statement:

```
SELECT * FROM [dbo].[DimCurrency]
WHERE [CurrencyAlternateKey]
IN ('ARS', 'AUD', 'BRL', 'CAD', 'CNY',
    'DEM', 'EUR', 'FRF', 'GBP', 'JPY',
    'MXN', 'SAR', 'USD', 'VEB')
```

7. On the **Columns** page, make the following selections:
   a. In the **Available Input Columns** panel, drag **CurrencyID** to the **Available Lookup Columns** panel and drop it on **CurrencyAlternateKey**.
   b. In the **Available Lookup Columns** list, select the check box to the left of **CurrencyKey**.
8. Click **OK** to return to the **Data Flow** design surface.
9. Right-click the Lookup Currency Key transformation, click **Properties**.
10. In the Properties window, verify that the **LocaleID** property is set to **English (United States)** and the **DefaultCodePage** property is set to **1252**.

To add and configure the Lookup DateKey transformation

1. In the **SSIS Toolbox**, drag **Lookup** onto the **Data Flow** design surface. Place Lookup directly below the **Lookup Currency Key** transformation.
2. Click the **Lookup Currency Key** transformation and drag the green arrow onto the newly added **Lookup** transformation to connect the two components.
3. In the **Input Output Selection** dialog box, click **Lookup Match Output** in the **Output** list box, and then click **OK**.
4. On the **Data Flow** design surface, click **Lookup** in the newly added **Lookup** transformation, and change the name to **Lookup Date Key**.
5. Double-click the **Lookup Date Key** transformation.
6. On the **General** page, select **Partial cache**.
7. On the **Connection** page, make the following selections:
   a. In the **OLEDB connection manager** dialog box, ensure that **localhost.AdventureWorksDW2012** is displayed.
   b. In the **Use a table or view** box, type or select **[dbo].[DimDate]**.
8. On the **Columns** page, make the following selections:
   a. In the **Available Input Columns** panel, drag **CurrencyDate** to the **Available Lookup Columns** panel and drop it on **FullDateAlternateKey**.
   b. In the **Available Lookup Columns** list, select the check box to the left of **DateKey**.
9. On the **Advanced** page, review the caching options.
10. Click **OK** to return to the **Data Flow** design surface.
11. Right-click the Lookup Date Key transformation and click **Properties.**
12. In the Properties window, verify that the **LocaleID** property is set to **English (United States)** and the **DefaultCodePage** property is set to **1252**.

Next Task in Lesson

Step 7: Adding and Configuring the OLE DB Destination

See Also

Lookup Transformation

Lesson 1-7 - Adding and Configuring the OLE DB Destination

10/1/2018 • 2 minutes to read • Edit Online

Your package now can extract data from the flat file source and transform that data into a format that is compatible with the destination. The next task is to actually load the transformed data into the destination. To load the data, you must add an OLE DB destination to the data flow. The OLE DB destination can use a database table, view, or an SQL command to load data into a variety of OLE DB-compliant databases.

In this procedure, you add and configure an OLE DB destination to use the OLE DB connection manager that you previously created.

To add and configure the sample OLE DB destination

1. In the **SSIS Toolbox**, expand **Other Destinations**, and drag **OLE DB Destination** onto the design surface of the **Data Flow** tab. Place the OLE DB destination directly below the **Lookup Date Key** transformation.
2. Click the **Lookup Date Key** transformation and drag the green arrow over to the newly added **OLE DB Destination** to connect the two components together.
3. In the **Input Output Selection** dialog box, in the **Output** list box, click **Lookup Match Output**, and then click **OK**.
4. On the **Data Flow** design surface, click **OLE DB Destination** in the newly added **OLE DB Destination** component, and change the name to **Sample OLE DB Destination**.
5. Double-click **Sample OLE DB Destination**.
6. In the **OLE DB Destination Editor** dialog box, ensure that **localhost.AdventureWorksDW2012** is selected in the **OLE DB Connection manager** box.
7. In the **Name of the table or the view** box, type or select **[dbo].[FactCurrencyRate]**.
8. Click the **New** button to create a new table. Change the name of the table in the script to read **NewFactCurrencyRate**. Click **OK**.
9. Upon clicking **OK**, the dialog will close and the **Name of the table or the view** will automatically change to **NewFactCurrencyRate**.
10. Click **Mappings**.
11. Verify that the **AverageRate**, **CurrencyKey**, **EndOfDayRate**, and **DateKey** input columns are mapped correctly to the destination columns. If same-named columns are mapped, the mapping is correct.
12. Click **OK**.
13. Right-click the **Sample OLE DB Destination** destination and click **Properties**.
14. In the Properties window, verify that the **LocaleID** property is set to **English (United States)** and the**DefaultCodePage** property is set to **1252**.

Next Task in Lesson

Step 8: Making the Lesson 1 Package Easier to Understand

See Also

OLE DB Destination

Lesson 1-8 - Making the Lesson 1 Package Easier to Understand

10/1/2018 • 2 minutes to read • Edit Online

Now that you have completed the configuration of the Lesson 1 package, it is a good idea to tidy up the package layout. If the shapes in the control and data flow layouts are random sizes, or if the shapes are not aligned or grouped, the functionality of package can be more difficult to understand.

SQL Server Data Tools provides tools that make it easy and quick to format the package layout. The formatting features include the ability to make shapes the same size, align shapes, and manipulate the horizontal and vertical spacing between shapes.

Another way to improve the understanding of package functionality is to add annotations that describe package functionality.

In this task, you will use the formatting features in SQL Server Data Tools to improve the layout of the data flow and also add an annotation to the data flow.

To format the layout of the data flow

1. If the Lesson 1 package is not already open, double-click Lesson 1.dtsx in Solution Explorer.
2. Click the **Data Flow** tab.
3. Place the cursor to the top and to the right of the Extract Sample Currency transformation, click, and then drag the cursor across all the data flow components.
4. On the **Format** menu, point to **Make Same Size**, and then click **Both**.
5. With the data flow objects selected, on the **Format** menu, point to **Align**, and then click **Lefts**.

To add an annotation to the data flow

1. Right-click anywhere in the background of the data flow design surface and then click **Add Annotation**.
2. Type or paste the following text in the annotation box.

   The data flow extracts data from a file, looks up values in the CurrencyKey column in the DimCurrency table and the DateKey column in the DimDate table, and writes the data to the NewFactCurrencyRate table.

   To wrap the text in the annotation box, place the cursor where you want to start a new line and press the Enter key.

   If you do not add text to the annotation box, it disappears when you click outside the box.

Next Steps

Step 9: Testing the Lesson 1 Tutorial Package

Lesson 1-9 - Testing the Lesson 1 Tutorial Package

10/1/2018 • 2 minutes to read • Edit Online

In this lesson, you have done the following tasks:

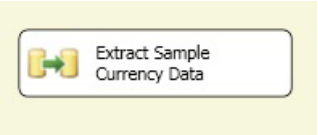
- Created a new SSIS project.
- Configured the connection managers that the package needs to connect to the source and destination data.
- Added a data flow that takes the data from a flat file source, performs the necessary Lookup transformations on the data, and configures the data for the destination.

Your package is now complete! It is time to test your package.

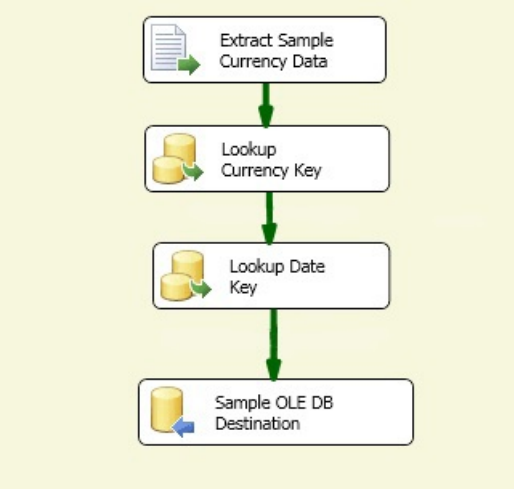
Checking the Package Layout

Before you test the package you should verify that the control and data flows in the Lesson 1 package contain the objects shown in the following diagrams.

Control Flow



Data Flow



To run the Lesson 1 tutorial package

1. On the **Debug** menu, click **Start Debugging**.

   The package will run, resulting in 1097 rows successfully added into the **FactCurrency** fact table in **AdventureWorksDW2012**.

2. After the package has completed running, on the **Debug** menu, click **Stop Debugging**.

Next Lesson

Lesson 2: Adding Looping with SSIS

See Also

Execution of Projects and Packages

Lesson 2: Adding Looping with SSIS

10/1/2018 • 2 minutes to read • Edit Online

In Lesson 1: Create a Project and Basic Package with SSIS, you created a package that extracted data from a single flat file source, transformed the data using Lookup transformations, and finally loaded the data into the **FactCurrency** fact table of the **AdventureWorksDW2012** sample database.

However, it is rare for an extract, transform, and load (ETL) process to use a single flat file. A typical ETL process would extract data from multiple flat file sources. Extracting data from multiple sources requires an iterative control flow. One of the most anticipated features of Microsoft Integration Services is the ability to easily add iteration or looping to packages.

Integration Services provides two types of containers for looping through packages: the Foreach Loop container and the For Loop container. The Foreach Loop container uses an enumerator to perform the looping, whereas the For Loop container typically uses a variable expression. This lesson uses the Foreach Loop container.

The Foreach Loop container enables a package to repeat the control flow for each member of a specified enumerator. With the Foreach Loop container, you can enumerate:

- ADO recordset rows
- ADO .Net schema information
- File and directory structures
- System, package and user variables
- Enumerable objects contained in a variable
- Items in a collection
- Nodes in an XML Path Language (XPath) expression
- SQL Server Management Objects (SMO)

In this lesson, you will modify the simple ETL package created in Lesson 1 to take advantage of the Foreach Loop container. You will also set user-defined package variables to enable the tutorial package to iterate through all the flat files in the folder. If you have not completed the previous lesson, you can also copy the completed Lesson 1 package that is included with the tutorial.

In this lesson, you will not modify the data flow, only the control flow.

> **IMPORTANT**
> This tutorial requires the **AdventureWorksDW2012** sample database. For more information about how to install and deploy **AdventureWorksDW2012**, see Reporting Services Product Samples on CodePlex.

Lesson Tasks

This lesson contains the following tasks:

- Step 1: Copying the Lesson 1 Package
- Step 2: Adding and Configuring the Foreach Loop Container

- Step 3: Modifying the Flat File Connection Manager
- Step 4: Testing the Lesson 2 Tutorial Package

Start the Lesson

Step 1: Copying the Lesson 1 Package

See Also

For Loop Container

Lesson 2-1 - Copying the Lesson 1 Package

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will create a copy of the Lesson 1.dtsx package that you created in Lesson 1. If you did not complete Lesson 1, you can add the completed lesson 1 package that is included with the tutorial to the project, and then copy it instead. You will use this new copy throughout the rest of Lesson 2.

To create the Lesson 2 package

1. If SQL Server Data Tools is not already open, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2012**, and then click **SQL Server Data Tools**.
2. On the **File** menu, click **Open**, click **Project/Solution**, click the **SSIS Tutorial** folder and click **Open**, and then double-click **SSIS Tutorial.sln**.
3. In Solution Explorer, right-click **Lesson 1.dtsx**, and then click **Copy**.
4. In Solution Explorer, right-click **SSIS Packages**, and then click **Paste**.

   By default, the copied package will be named Lesson 2.dtsx.
5. In Solution Explorer, double-click **Lesson 2.dtsx** to open the package
6. Right-click anywhere in the background of the **Control Flow** design surface and click **Properties**.
7. In the Properties window, update the **Name** property to **Lesson 2**.
8. Click the box for the **ID** property, click the dropdown arrow and then click .

To add the completed Lesson 1 package

1. Open SQL Server Data Tools and open the SSIS Tutorial project.
2. In Solution Explorer, right-click **SSIS Packages**, and click **Add Existing Package**.
3. In the **Add Copy of Existing Package** dialog box, in **Package location**, select **File system**.
4. Click the browse **(...)** button, navigate to **Lesson 1.dtsx** on your machine, and then click **Open**.

   To download all of the lesson packages for this tutorial, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Copy and paste the Lesson 1 package as described in steps 3-8 in the previous procedure.

Next Task in Lesson

Step 2: Adding and Configuring the Foreach Loop Container

Lesson 2-2 - Adding and Configuring the Foreach Loop Container

10/1/2018 • 3 minutes to read • Edit Online

In this task, you will add the ability to loop through a folder of flat files and apply the same data flow transformation used in Lesson 1 to each of those flat files. You do this by adding and configuring a Foreach Loop container to the control flow.

The Foreach Loop container that you add must be able to connect to each flat file in the folder. Because all the files in the folder have the same format, the Foreach Loop container can use the same Flat File connection manager to connect to each of these files. The Flat File connection manager that the container will use is the same Flat File connection manager that you created in Lesson 1.

Currently, the Flat File connection manager from Lesson 1 connects to only one, specific flat file. To iteratively connect to each flat file in the folder, you will have to configure both the Foreach Loop container and the Flat File connection manager as follows:

- **Foreach Loop container:** You will map the enumerated value of the container to a user-defined package variable. The container will then use this user-defined variable to dynamically modify the **ConnectionString** property of the Flat File connection manager and iteratively connect to each flat file in the folder.
- **Flat File connection manager:** You will modify the connection manager that was created in Lesson 1 by using a user-defined variable to populate the connection manager's **ConnectionString** property.

The procedures in this task show you how to create and modify the Foreach Loop container to use a user-defined package variable and to add the data flow task to the loop. You will learn how to modify the Flat File connection manager to use a user-defined variable in the next task.

After you have made these modifications to the package, when the package is run, the Foreach Loop Container will iterate through the collection of files in the Sample Data folder. Each time a file is found that matches the criteria, the Foreach Loop Container will populate the user-defined variable with the file name, map the user-defined variable to the **ConnectionString** property of the Sample Currency Data Flat File connection manager, and then run the data flow against that file. Therefore, in each iteration of the Foreach Loop the Data Flow task will consume a different flat file.

> **NOTE**
>
> Because Microsoft Integration Services separates control flow from data flow, any looping that you add to the control flow will not require modification to the data flow. Therefore, the data flow that you created in Lesson 1 does not have to be changed.

To add a Foreach Loop container

1. In **SQL Server Data Tools**, click the **Control Flow** tab.
2. In the **SSIS Toolbox**, expand **Containers**, and then drag a **Foreach Loop Container** onto the design surface of the **Control Flow** tab.
3. Right-click the newly added **Foreach Loop Container** and select **Edit**.
4. In the **Foreach Loop Editor** dialog box, on the **General** page, for **Name**, enter **Foreach File in Folder**.

Click **OK**.

5. Right-click the Foreach Loop container, click **Properties**, and in the Properties window, verify that the **LocaleID** property is set to **English (United States)**.

To configure the enumerator for the Foreach Loop container

1. Double-click Foreach File in Folder to reopen the **Foreach Loop Editor**.
2. Click **Collection**.
3. On the **Collection** page, select **Foreach File Enumerator**.
4. In the **Enumerator configuration** group, click **Browse**.
5. In the **Browse for Folder** dialog box, locate the folder on your machine that contains the Currency_*.txt files.

   This sample data is included with the SSIS lesson packages. To download the sample data and the lesson packages, do the following.

   a. Navigate to Integration Services Product Samples.

   b. Click the **DOWNLOADS** tab.

   c. Click the link for the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
6. In the **Files** box, type **Currency_*.txt**.

To map the enumerator to a user-defined variable

1. Click **Variable Mappings**.
2. On the **Variable Mappings** page, in the **Variable** column, click the empty cell and select **<New Variable...>**.
3. In the **Add Variable** dialog box, for **Name**, type **varFileName**.

   > **IMPORTANT**
   > Variable names are case sensitive.

4. Click **OK**.
5. Click **OK** again to exit the **Foreach Loop Editor** dialog box.

To add the data flow task to the loop

- Drag the **Extract Sample Currency Data** data flow task onto the Foreach Loop container now renamed **Foreach File in Folder**.

Next Lesson Task

Step 3: Modifying the Flat File Connection Manager

See Also

Configure a Foreach Loop Container
Use Variables in Packages

Lesson 2-3 - Modifying the Flat File Connection Manager

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will modify the Flat File connection manager that you created and configured in Lesson 1. When originally created, the Flat File connection manager was configured to statically load a single file. To enable the Flat File connection manager to iteratively load files, you must modify the ConnectionString property of the connection manager to accept the user-defined variable `User:varFileName`, which contains the path of the file to be loaded at run time.

By modifying the connection manager to use the value of the user-defined variable, `User::varFileName`, to populate the ConnectionString property of the connection manager, the connection manager will be able to connect to different flat files. At run time, each iteration of the Foreach Loop container will dynamically update the `User::varFileName` variable. Updating the variable, in turn, causes the connection manager to connect to a different flat file, and the data flow task to process a different set of data.

To configure the Flat File connection manager to use a variable for the connection string

1. In the **Connection Managers** pane, right-click **Sample Flat File Source Data**, and select **Properties**.
2. In the Properties window, for **Expressions**, click in the empty cell, and then click the ellipsis button **(...)**.
3. In the **Property Expressions Editor** dialog box, in the **Property** column, type or select **ConnectionString**.
4. In the **Expression** column, click the ellipsis button **(...)** to open the **Expression Builder** dialog box.
5. In the **Expression Builder** dialog box, expand the **Variables** node.
6. Drag the variable, **User::varFileName**, into the **Expression** box.
7. Click **OK** to close the **Expression Builder** dialog box.
8. Click **OK** again to close the **Property Expressions Editor** dialog box.

Next Lesson Task

Step 4: Testing the Lesson 2 Tutorial Package

Lesson 2-4 - Testing the Lesson 2 Tutorial Package

10/1/2018 • 2 minutes to read • Edit Online

With the Foreach Loop container and the Flat File connection manager now configured, the Lesson 2 package can iterate through the collection of 14 flat files in the Sample Data folder. Each time a file name is found that matches the specified file name criteria, the Foreach Loop container populates the user-defined variable with the file name. This variable, in turn, updates the ConnectionString property of the Flat File connection manager, and a connection to the new flat file is made. The Foreach Loop container then runs the unmodified data flow task against the data in the new flat file before connecting to the next file in the folder.

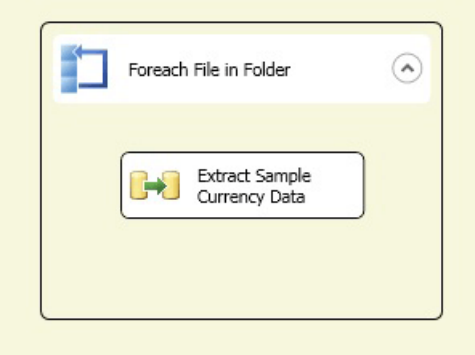
Use the following procedure to test the new looping functionality that you have added to your package.

> **NOTE**
> If you ran the package from Lesson 1, you will need to delete the records from dbo.FactCurrency in AdventureWorksDW2012 before you run the package from this lesson or the package will fail with errors indicating a Violation of Primary Key constraint. You will receive the same errors if you run the package by selecting Debug/Start Debugging (or press F5) because both Lesson 1 and Lesson 2 will run. Lesson 2 will attempt to insert records already inserted in Lesson 1.

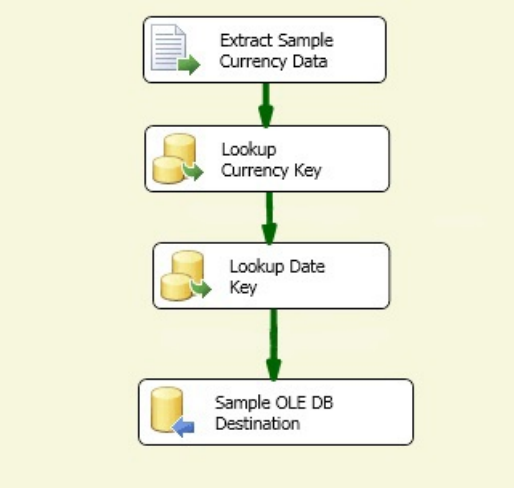
Checking the Package Layout

Before you test the package you should verify that the control and data flows in the Lesson 2 package contains the objects shown in the following diagrams. The data flow should be identical to the data flow in lesson 1.

Control Flow



Data Flow



To test the Lesson 2 tutorial package

1. In **Solution Explorer**, right-click **Lesson 2.dtsx** and click **Execute Package**.

   The package will run. You can verify the status of each loop in the Output window, or by clicking on the **Progress** tab. For example, you can see that 1097 lines were added to the destination table from the file Currency_VEB.txt.

2. After the package has completed running, on the **Debug** menu, click **Stop Debugging**.

Next Lesson

Lesson 5: Add SSIS Package Configurations for the Package Deployment Model

See Also

Execution of Projects and Packages

Lesson 3: Add Logging with SSIS

10/1/2018 • 2 minutes to read • Edit Online

Microsoft Integration Services includes logging features that let you troubleshoot and monitor package execution by providing a trace of task and container events. The logging features are flexible, and can be enabled at the package level or on individual tasks and containers within the package. You can select which events you want to log, and create multiple logs against a single package.

Logging is provided by a log provider. Each log provider can write logging information to different formats and destination types. Integration Services provides the following log providers:

- Text file
- SQL Server Profiler
- Windows Event log
- SQL Server
- XML file

In this lesson, you will create a copy of the package that you created in Lesson 2: Adding Looping with SSIS. Working with this new package, you will then add and configure logging to monitor specific events during package execution. If you have not completed any of the previous lessons, you can also copy the completed Lesson 2 package that is included with the tutorial.

> **IMPORTANT**
> This tutorial requires the **AdventureWorksDW2012** sample database. For more information about how to install and deploy **AdventureWorksDW2012**, Reporting Services Product Samples on CodePlex

Lesson Tasks

This lesson contains the following tasks:

- Step 1: Copying the Lesson 2 Package
- Step 2: Adding and Configuring Logging
- Step 3: Testing the Lesson 3 Tutorial Package

Start the Lesson

Step 1: Copying the Lesson 2 Package

Lesson 3-1 - Copying the Lesson 2 Package

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will create a copy of the Lesson 2.dtsx package that you created in Lesson 2. Alternatively, you can add the completed lesson 2 package that is included with the tutorial to the project, and then copy it instead. You will use this new copy throughout the rest of Lesson 3.

To create the Lesson 3 package

1. If SQL Server Data Tools is not already open, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2012**, and then click **SQL Server Data Tools**.
2. On the **File** menu, click **Open**, click **Project/Solution**, select **SSIS Tutorial** and click **Open**, and then double-click **SSIS Tutorial.sln**.
3. In Solution Explorer, right-click **Lesson 2.dtsx**, and then click **Copy**.
4. In Solution Explorer, right-click **SSIS Packages**, and then click **Paste**.

   By default, the copied package is named Lesson 3.dtsx.
5. In Solution Explorer, double-click **Lesson 3.dtsx** to open the package.
6. Right-click anywhere in the background of the **Control Flow** tab and click **Properties**.
7. In the Properties window, update the **Name** property to **Lesson 3**.
8. Click the box for the **ID** property, and then in the list, click .

To add the completed Lesson2 package

1. Open SQL Server Data Tools (SSDT) and open the SSIS Tutorial project.
2. In Solution Explorer, right-click **SSIS Packages**, and click **Add Existing Package**.
3. In the **Add Copy of Existing Package** dialog box, in **Package location**, select **File system**.
4. Click the browse **(...)** button, navigate to **Lesson 2.dtsx** on your machine, and then click **Open**.

   To download all of the lesson packages for this tutorial, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Copy and paste the Lesson 3 package as described in steps 3-8 in the previous procedure.

Next Task in Lesson

Step 2: Adding and Configuring Logging

Lesson 3-2 - Adding and Configuring Logging

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will enable logging for the data flow in the Lesson 3.dtsx package. Then, you will configure a Text File log provider to log the PipelineExecutionPlan and PipelineExecuteTrees events. The Text Files log provider creates logs that are easy to view and easily transportable. The simplicity of these log files makes these files especially useful during the basic testing phase of a package. You can also view the log entries in the Log Events window of SSIS Designer.

To add logging to the package

1. On the **SSIS** menu, click **Logging**.
2. In the **Configure SSIS Logs** dialog box, in the **Containers** pane, make sure that the topmost object, which represents the Lesson 3 package, is selected.
3. On the **Providers and Logs** tab, in the **Provider type** box, select **SSIS log provider for Text files**, and then click **Add**.

   Integration Services adds a new Text File log provider to the package with the default name **SSIS log provider for text files**. You can now configure the new log provider.
4. In the **Name** column, type **Lesson 3 Log File**.
5. Optionally, modify the **Description**.
6. In the **Configuration** column, click to specify the destination to which the log information is written.

   In the **File Connection Manager Editor** dialog box, for **Usage type**, select **Create file**, and then click **Browse**. By default, the **Select File** dialog box opens the project folder, but you can save log information to any location.
7. In the **Select File** dialog box, in the **File name** box type **TutorialLog.log**, and click **Open**.
8. Click **OK** to close the **File Connection Manager Editor** dialog box.
9. In the **Containers** pane, expand all nodes of the package container hierarchy, and then clear all check boxes, including the **Extract Sample Currency Data** check box. Now select the check box for **Extract Sample Currency Data** to get only the events for this node.

   > **IMPORTANT**
   > If the state of the **Extract Sample Currency Data** check box is dimmed instead of selected, the task uses the log settings of the parent container and you cannot enable the log events that are specific to the task.
10. On the **Details** tab, in the **Events** column, select the **PipelineExecutionPlan** and **PipelineExecutionTrees** events.
11. Click **Advanced** to review the details that the log provider will write to the log for each event. By default, all information categories are automatically selected for the events you specify.
12. Click **Basic** to hide the information categories.
13. On the **Provider and Logs** tab, in the **Name** column, select **Lesson 3 Log File**. Once you have created a log provider for your package, you can optionally deselect it to temporarily turn off logging, without having

to delete and re-create a log provider.

14. Click **OK**.

Next Steps

Step 3: Testing the Lesson 3 Tutorial Package

Lesson 3-3 - Testing the Lesson 3 Tutorial Package

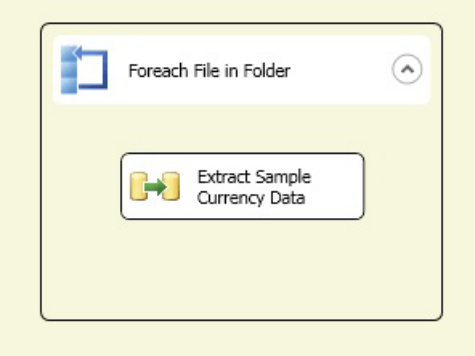
10/1/2018 • 2 minutes to read • Edit Online

In this task, you will run the Lesson 3.dtsx package. When the package runs, the Log Events window will list the log entries that are written to the log file. After the package finishes execution, you will then verify the contents of the log file that was generated by the log provider.

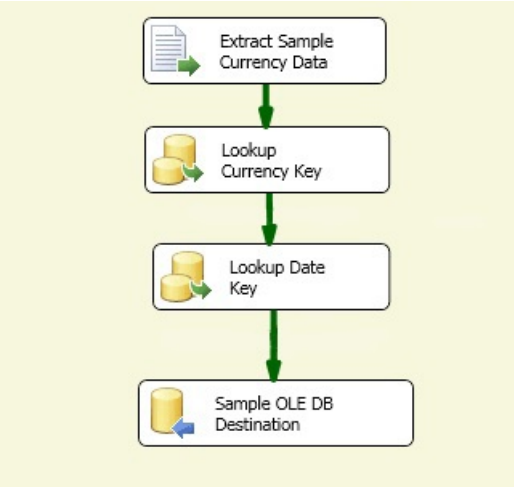
Checking the Package Layout

Before you test the package you should verify that the control and data flows in the Lesson 3 package contain the objects shown in the following diagrams. The control flow should be identical to the control flow in lesson 2. The data flow should be identical to the data flow in lessons 1 and 2.

Control Flow



Data Flow



To run the Lesson 4 tutorial package

1. On the SSIS menu, click Log Events.
2. On **Debug** menu, click **Start Debugging**.
3. After the package has completed running, on the **Debug** menu, click **Stop Debugging**.

To examine the generated log file

- Using Notepad or any other text editor, open the TutorialLog.log file.
- Although the semantics of the information generated for the **PipelineExecutionPlan** and **PipelineExecutionTrees** events are beyond the scope of this tutorial, you can see that the first line lists the

information fields specified in the **Details** tab of the **Configure SSIS Logs** dialog box. Moreover, you can verify that the two events that you selected, PipelineExecutionPlan and PipelineExecutionTrees, have been logged for each iteration of the Foreach Loop.

Next Lesson

Lesson 4: Add Error Flow Redirection with SSIS

Lesson 4: Add Error Flow Redirection with SSIS

10/1/2018 • 2 minutes to read • Edit Online

To handle errors that may occur in the transformation process, Microsoft Integration Services gives you the ability to decide on a per component and per column basis how to handle data that cannot be transformed. You can choose to ignore a failure in certain columns, redirect the entire failed row, or just fail the component. By default, all components in Integration Services are configured to fail when errors occur. Failing a component, in turn, causes the package to fail and all subsequent processing to stop.

Instead of letting failures stop package execution, it is good practice to configure and handle potential processing errors as they occur within the transformation. While you might choose to ignore failures to ensure your package runs successfully, it is often better to redirect the failed row to another processing path where the data and the error can be persisted, examined and reprocessed at a later time.

In this lesson, you will create a copy of the package that you developed in Lesson 3: Add Logging with SSIS. Working with this new package, you will create a corrupted version of one of the sample data files. The corrupted file will force a processing error to occur when you run the package.

To handle the error data, you will add and configure a Flat File destination that will write any rows that fail to locate a lookup value in the Lookup Currency Key transformation to a file.

Before the error data is written to the file, you will include a Script component that uses script to get error descriptions. You will then reconfigure the Lookup Currency Key transformation to redirect any data that could not be processed to the Script transformation.

> **IMPORTANT**
> This tutorial requires the **AdventureWorksDW2012** sample database. For more information about how to install and deploy **AdventureWorksDW2012**, Reporting Services Product Samples on CodePlex

Tasks in Lesson

This lesson contains the following tasks:

- Step 1: Copying the Lesson 3 Package
- Step 2: Creating a Corrupted File
- Step 3: Adding Error Flow Redirection
- Step 4: Adding a Flat File Destination
- Step 5: Testing the Lesson 4 Tutorial Package

Start the Lesson

Step 1: Copying the Lesson 3 Package

Lesson 4-1 - Copying the Lesson 3 Package

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will create a copy of the Lesson 3.dtsx package that you created in Lesson 3. Alternatively, if you did not complete lesson 3, you can add the completed lesson 3 package that is included with the tutorial to the project, and then make a copy of it to work with. You will use this new copy throughout the rest of Lesson 4.

To create the Lesson 4 package

1. If SQL Server Data Tools is not already open, click **Start**, point to **All Programs**, point to **Microsoft SQL Server**, and then click **SQL Server Data Tools**.
2. On the **File** menu, click **Open**, click **Project/Solution**, select **SSIS Tutorial** and click **Open**, and then double-click **SSIS Tutorial.sln**.
3. In Solution Explorer, right-click **Lesson 3.dtsx**, and then click **Copy**.
4. In Solution Explorer, right-click **SSIS Packages**, and then click **Paste**.

   By default, the copied package is named Lesson 4.dtsx.
5. In Solution Explorer, double-click **Lesson 4.dtsx** to open the package.
6. Right-click anywhere in the background of the **Control Flow** tab and click **Properties**.
7. In the Properties window, update the **Name** property to **Lesson 4**.
8. Click the box for the **ID** property, and then in the list, click .

To add the completed Lesson 3 package

1. Open SQL Server Data Tools (SSDT) and open the SSIS Tutorial project.
2. In Solution Explorer, right-click **SSIS Packages**, and click **Add Existing Package**.
3. In the **Add Copy of Existing Package** dialog box, in **Package location**, select **File system**.
4. Click the browse **(...)** button, navigate to Lesson 3.dtsx on your machine, and then click **Open**.

   To download all of the lesson packages for this tutorial, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Copy and paste the Lesson 3 package as described in steps 3-8 in the previous procedure.

Next Task in Lesson

Step 2: Creating a Corrupted File

Lesson 4-2 - Creating a Corrupted File

10/1/2018 • 2 minutes to read • Edit Online

In order to demonstrate the configuration and handling of transformation errors, you will have to create a sample flat file that when processed causes a component to fail.

In this task, you will create a copy of an existing sample flat file. You will then open the file in Notepad and edit the **CurrencyID** column to ensure that it will fail to produce a match during the transformations lookup. When the new file is processed, the lookup failure will cause the Currency Key Lookup transformation to fail and therefore fail the rest of the package. After you have created the corrupted sample file, you will run the package to view the package failure.

To create a corrupted sample flat file

1. In Notepad or any other text editor, open the Currency_VEB.txt file.

   The sample data is included with the SSIS Lesson packages. To download the sample data and the lesson packages, do the following.

   a. Navigate to Integration Services Product Samples.

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.

2. Use the text editor's find and replace feature to find all instances of **VEB** and replace them with **BAD**.

3. In the same folder as the other sample data files, save the modified file as **Currency_BAD.txt**.

   > **IMPORTANT**
   > Make sure that **Currency_BAD.txt** is saved the same folder as the other sample data files.

4. Close your text editor.

To verify that an error will occur during run time

1. On the **Debug** menu, click **Start Debugging**.

   On the third iteration of the data flow, the Lookup Currency Key transformation tries to process the Currency_BAD.txt file, and the transformation will fail. The failure of the transformation will cause the whole package to fail.

2. On the **Debug** menu, click **Stop Debugging**.

3. On the design surface, click the **Execution Results** tab.

4. Browse through the log and verify that the following unhandled error occurred:

   ```
   [Lookup Currency Key[27]] Error: Row yielded no match during lookup.
   ```

   > **NOTE**
   > The number 27 is the ID of the component. This value is assigned when you build the data flow, and the value in your package may be different.

Next Steps

Step 3: Adding Error Flow Redirection

Lesson 4-3 - Adding Error Flow Redirection

10/1/2018 • 2 minutes to read • Edit Online

As demonstrated in the previous task, the Lookup Currency Key transformation cannot generate a match when the transformation tries to process the corrupted sample flat file, which produced an error. Because the transformation uses the default settings for error output, any error causes the transformation to fail. When the transformation fails, the rest of the package also fails.

Instead of permitting the transformation to fail, you can configure the component to redirect the failed row to another processing path by using the error output. Use of a separate error processing path lets you do a number of things. For instance, you might try to clean the data and then reprocess the failed row. Or, you might save the failed row along with additional error information for later verification and reprocessing.

In this task, you will configure the Lookup Currency Key transformation to redirect any rows that fail to the error output. In the error branch of the data flow, these rows will be written to a file.

By default the two extra columns in an Integration Services error output, **ErrorCode** and **ErrorColumn**, contain only numeric codes that represent an error number, and the ID of the column in which the error occurred. These numeric values may be of limited use without the corresponding error description.

To enhance the usefulness of the error output, before the package writes the failed rows to the file, you will use a Script component to access the Integration Services API and get a description of the error.

To configure an error output

1. In the **SSIS Toolbox**, expand **Common**, and then drag **Script Component** onto the design surface of the **Data Flow** tab. Place **Script** to the right of the **Lookup Currency Key** transformation.
2. In the **Select Script Component Type** dialog box, click **Transformation**, and click **OK**.
3. Click the **Lookup Currency Key** transformation and then drag the red arrow onto the newly added **Script** transformation to connect the two components.

   The red arrow represents the error output of the **Lookup Currency Key** transformation. By using the red arrow to connect the transformation to the Script component, you can redirect any processing errors to the Script component, which then processes the errors and sends them to the destination.
4. In the **Configure Error Output** dialog box, in the **Error** column, select **Redirect row**, and then click **OK**.
5. On the **Data Flow** design surface, click **Script Component** in the newly added **ScriptComponent**, and change the name to **Get Error Description**.
6. Double-click the **Get Error Description** transformation.
7. In the **Script Transformation Editor** dialog box, on the **Input Columns** page, select the **ErrorCode** column.
8. On the **Inputs and Outputs** page, expand **Output 0**, click **Output Columns**, and then click **Add Column**.
9. In the **Name** property, type **ErrorDescription** and set the **DataType** property to **Unicode string [DT_WSTR]**.
10. On the **Script** page, verify that the **LocaleID** property is set to **English (United States.**
11. Click **Edit Script** to open Microsoft Visual Studio Tools for Applications (VSTA). In the

Input0_ProcessInputRow method, type or paste the following code.

[Visual Basic]

```
Row.ErrorDescription =
  Me.ComponentMetaData.GetErrorDescription(Row.ErrorCode)
```

[Visual C#]

```
Row.ErrorDescription = this.ComponentMetaData.GetErrorDescription(Row.ErrorCode);
```

The completed subroutine will look like the following code.

[Visual Basic]

```
Public Overrides Sub Input0_ProcessInputRow(ByVal Row As Input0Buffer)

  Row.ErrorDescription =
    Me.ComponentMetaData.GetErrorDescription(Row.ErrorCode)

End Sub
```

[Visual C#]

```
public override void Input0_ProcessInputRow(Input0Buffer Row)
    {

        Row.ErrorDescription = this.ComponentMetaData.GetErrorDescription(Row.ErrorCode);

    }
```

12. On the **Build** menu, click **Build Solution** to build the script and save your changes, and then close VSTA.
13. Click **OK** to close the **Script Transformation Editor** dialog box.

Next Steps

Step 4: Adding a Flat File Destination

Lesson 4-4 - Adding a Flat File Destination

10/1/2018 • 2 minutes to read • Edit Online

The error output of the Lookup Currency Key transformation redirects to the Script transformation any data rows that failed the lookup operation. To enhance information about the errors that occurred, the Script transformation runs a script that gets the description of errors.

In this task, you will save all this information about the failed rows to a delimited file for later processing. To save the failed rows, you must add and configure a Flat File connection manager for the text file that will contain the error data and a Flat File destination. By setting properties on the Flat File connection manager that the Flat File destination uses, you can specify how the Flat File destination formats and writes the text file. For more information, see Flat File Connection Manager and Flat File Destination.

To add and configure a Flat File destination

1. Click the **Data Flow** tab.
2. In the **SSIS Toolbox**, expand **Other**, and drag **Flat File Destination** onto the data flow design surface. Put the **Flat File Destination** directly underneath the **Get Error Description** transformation.
3. Click the **Get Error Description** transformation, and then drag the green arrow onto the new **Flat File Destination**.
4. On the **Data Flow** design surface, click **Flat File Destination** in the newly added **Flat File Destination** transformation, and change the name to **Failed Rows**.
5. Right-click the **Failed Rows** transformation, click **Edit**, and then in the **Flat File Destination Editor**, click **New**.
6. In the **Flat File Format** dialog box, verify that **Delimited** is selected, and then click **OK**.
7. In the **Flat File Connection Manager Editor**, in the **Connection Manager Name** box type **Error Data**.
8. In the **Flat File Connection Manager Editor** dialog box, click **Browse**, and locate the folder in which to store the file.
9. In the **Open** dialog box, for **File name**, type **ErrorOutput.txt**, and then click **Open**.
10. In the **Flat File Connection Manager Editor** dialog box, verify that the **Locale** box contains English (United States) and **Code page** contains 1252 (ANSI -Latin I).
11. In the options pane, click **Columns**.

    Notice that, in addition to the columns from the source data file, three new columns are present: ErrorCode, ErrorColumn, and ErrorDescription. These columns are generated by the error output of the Lookup Currency Key transformation and by the script in the Get Error Description transformation, and can be used to troubleshoot the cause of the failed row.
12. Click **OK**.
13. In the **Flat File Destination Editor**, clear the **Overwrite data in the file** check box.

    Clearing this check box persists the errors over multiple package executions.
14. In the **Flat File Destination Editor**, click **Mappings** to verify that all the columns are correct. Optionally, you can rename the columns in the destination.

15. Click **OK**.

Next Steps

Step 5: Testing the Lesson 4 Tutorial Package

Lesson 4-5 - Testing the Lesson 4 Tutorial Package

10/1/2018 • 2 minutes to read • Edit Online

At run time, the corrupted file, Currency_BAD.txt, will fail to generate a match within the Currency Key Lookup transformation. Because the error output of Currency Key Lookup has now been configured to redirect failed rows to the new Failed Rows destination, the component does not fail, and the package runs successfully. All failed error rows are written to ErrorOutput.txt.

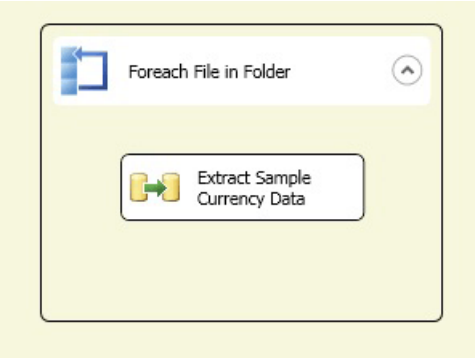
In this task, you will test the revised error output configuration by running the package. Upon successful package execution, you will then view the contents of the ErrorOutput.txt file.

> **NOTE**
> If you do not want to accumulate error rows in the ErrorOutput.txt file, you should manually delete the file content between package runs.

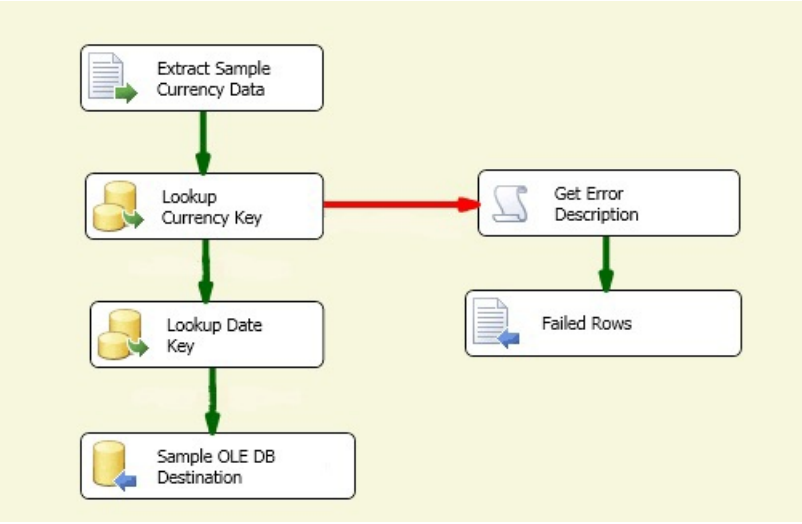
Checking the Package layout

Before you test the package you should verify that the control flow and the data flow in the Lesson 4 package contain the objects shown in the following diagrams. The control flow should be identical to the control flow in lessons 2 - 4.

Control Flow



Data Flow



To run the Lesson 4 tutorial package

1. On the **Debug** menu, click **Start Debugging**.
2. After the package has completed running, on the **Debug** menu, click **Stop Debugging**.

To verify the contents of the ErrorOutput.txt file

- In Notepad or any other text editor, open the ErrorOutput.txt file. The default column order is: AverageRate, CurrencyID, CurrencyDate, EndOfDateRate, ErrorCode, ErrorColumn, ErrorDescription.

  Notice that all the rows in the file contain the unmatched CurrencyID value of BAD, the ErrorCode value of -1071607778, the ErrorColumn value of 0, and the ErrorDescription value "Row yielded no match during lookup". The value of the ErrorColumn is set to 0 because the error is not column specific. It is the lookup operation that failed. .

Lesson 5: Add SSIS Package Configurations for the Package Deployment Model

10/1/2018 • 2 minutes to read • Edit Online

Package configurations let you set run-time properties and variables from outside of the development environment. Configurations allow you to develop packages that are flexible and easy to both deploy and distribute. Microsoft Integration Services offers the following configuration types:

- XML configuration file
- Environment variable
- Registry entry
- Parent package variable
- SQL Server table

In this lesson, you will modify the simple Integration Services package that you created in Lesson 4: Add Error Flow Redirection with SSIS to use the Package Deployment Model and take advantage of package configurations. You can also copy the completed Lesson 4 package that is included with the tutorial. Using the Package Configuration Wizard, you will create an XML configuration that updates the **Directory** property of the Foreach Loop container by using a package-level variable mapped to the Directory property. Once you have created the configuration file, you will modify the value of the variable from outside of the development environment and point the modified property to a new sample data folder. When you run the package again, the configuration file populates the value of the variable, and the variable in turn updates the **Directory**property. As a result, the package iterates through the files in the new data folder, rather than iterating through the files in the original folder that was hard-coded in the package.

> **IMPORTANT**
> This tutorial requires the **AdventureWorksDW2012** sample database. For more information about how to install and deploy **AdventureWorksDW2012**, see Reporting Services Product Samples on CodePlex.

Lesson Tasks

This lesson contains the following tasks:

- Step 1: Copying the Lesson 4 Package
- Step 2: Enabling and Configuring Package Configurations
- Step 3: Modifying the Directory Property Configuration Value
- Step 4: Testing the Lesson 5 Tutorial Package

Start the Lesson

- Step 1: Copying the Lesson 4 Package

Lesson 5-1 - Copying the Lesson 4 Package

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will create a copy of the Lesson 4.dtsx package that you created in Lesson 4. Alternatively, you can add the completed lesson 4 package that is included with the tutorial to the project, and then copy it instead. You will use this new copy throughout the rest of Lesson 5.

To copy the Lesson 4 package

1. If SQL Server Data Tools is not already open, click **Start**, point to **All Programs**, point to **Microsoft SQL Server 2012**, and then click **SQL Server Data Tools**.
2. On the **File** menu, click **Open**, click **Project/Solution**, select **SSIS Tutorial** and click **Open**, and then double-click **SSIS Tutorial.sln**.
3. In Solution Explorer, right-click **Lesson 4.dtsx**, and then click **Copy**.
4. In Solution Explorer, right-click **SSIS Packages**, and then click **Paste**.

   By default, the copied package is named Lesson 5.dtsx.
5. In the Solution Explorer, double-click **Lesson 5.dtsx** to open the package.
6. Right-click anywhere in the background of the **Control Flow** tab then click **Properties**.
7. In the Properties window, update the **Name** property to **Lesson 5**.
8. Click the box for the **ID** property, then click the dropdown arrow, and then click .

To add the completed Lesson 4 package

1. Open SQL Server Data Tools and open the SSIS Tutorial project.
2. In Solution Explorer, right-click **SSIS Packages**, and click **Add Existing Package**.
3. In the **Add Copy of Existing Package** dialog box, in **Package location**, select **File system**.
4. Click the browse **(...)** button, navigate to **Lesson 4.dtsx** on your machine, and then click **Open**.

   To download all of the lesson packages for this tutorial, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Copy and paste the Lesson 4 package as described in steps 3-8 in the previous procedure.

Next Task in Lesson

Step 2: Enabling and Configuring Package Configurations

Lesson 5-2 - Enabling and Configuring Package Configurations

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will convert the project to the Package Deployment Model and enable package configurations using the Package Configuration Wizard. You will use this wizard to generate an XML configuration file that contains configuration settings for the **Directory** property of the Foreach Loop container. The value of the Directory property is supplied by a new package-level variable that you can update at run time. Additionally, you will populate a new sample data folder to use during testing.

To create a new package-level variable mapped to the Directory property

1. Click the background of the **Control Flow** tab in SSIS Designer. This sets the scope for the variable you will create to the package.
2. On the SSIS menu, select **Variables**.
3. In the **Variables** window, click the Add Variable icon.
4. In the **Name** box, type **varFolderName**.

   > **IMPORTANT**
   > Variable names are case sensitive.

5. Verify that the **Scope** box shows the name of the package, Lesson 5.
6. Set the value of the **Data Type** box of the `varFolderName` variable to **String**.
7. Return to the **Control Flow** tab and double-click the **Foreach File in Folder** container.
8. On the **Collection** page of the **Foreach Loop Editor**, click **Expressions**, and then click the ellipsis button **(...)**.
9. In the **Property Expressions Editor**, click in the **Property** list, and select **Directory**.
10. In the **Expression** box, click the ellipsis button**(...)**.
11. In the **Expression Builder**, expand the Variables folder, and drag the variable **User::varFolderName** to the **Expression** box.
12. Click **OK** to exit the **Expression Builder**.
13. Click **OK** to exit the **Property Expressions Editor**.
14. Click **OK** to exit the **Foreach Loop Editor**.

To enable package configurations

1. On the **Project Menu**, click **Convert to Package Deployment Model**.
2. Click **OK** on the warning prompt and, once the conversion is complete, click **OK** on the **Convert to Package Deployment Model** dialog.
3. Click the background of the **Control Flow** tab in SSIS Designer.

4. On the **SSIS** menu, click **Package Configurations**.
5. In the **Package Configurations Organizer** dialog box, select **Enable Package Configurations**, and then click **Add**.
6. On the welcome page of the Package Configuration Wizard, click **Next**.
7. On the **Select Configuration Type** page, verify that the **Configuration type** is set to **XML configuration file**.
8. On the **Select Configuration Type** page, click **Browse**.
9. By default, the **Select Configuration File Location** dialog box will open to the project folder.
10. In the **Select Configuration File Location** dialog box, for **File name** type **SSISTutorial**, and then click **Save**.
11. On the **Select Configuration Type** page, click **Next.**
12. On the **Select Properties to Export** page, in the **Objects** pane, expand **Variables**, expand **varFolderName**, expand **Properties**, and then select **Value**.
13. On the **Select Properties to Export** page, click **Next**.
14. On the **Completing the Wizard** page, type a configuration name for the configuration, such as **SSIS Tutorial Directory configuration**. This is the configuration name that is displayed in the **Package Configuration Organizer** dialog box.
15. Click **Finish**.
16. Click **Close**.
17. The wizard creates a configuration file, named SSISTutorial.dtsConfig, that contains configuration settings for the v**alue** of the variable that in turn sets the **Directory** property of the enumerator.

> **NOTE**
> A configuration file typically contains complex information about the package properties, but for this tutorial the only configuration information should be
> <Configuration ConfiguredType="Property"
> Path="\Package.Variables[User::varFolderName].Properties[Value]" ValueType="String">
> </ConfiguredValue>
> </Configuration>.

To create and populate a new sample data folder

1. In Windows Explorer, at the root level of your drive (for example, C:\), create a new folder named **New Sample Data**.
2. Locate the sample files on your computer and copy three of the files from the folder.
3. In the **New Sample Data** folder, paste the copied files.

Next Task in Lesson

Step 3: Modifying the Directory Property Configuration Value

Lesson 5-3 - Modifying the Directory Property Configuration Value

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will modify the configuration setting, stored in the SSISTutorial.dtsConfig file, for the Value property of the package-level variable `User::varFolderName`. The variable updates the Directory property of the Foreach Loop container. The modified value will point to the **New Sample Data** folder that you created in the previous task. After you modify the configuration setting and run the package, the Directory property will be updated by the variable, using the value populated from the configuration file instead of the directory value originally configured in the package.

To modify the configuration setting of the Directory property

1. In Notepad or any other text editor, locate and open the SSISTutorial.dtsConfig configuration file that you created by using the Package Configuration Wizard in the previous task.

2. Change the value of the **ConfiguredValue** element to match the path of the **New Sample Data** folder that you created in the previous task. Do not surround the path in quotes. If the **New Sample Data** folder is at the root level of your drive (for example, C:\), the updated XML should be similar to the following sample:

   ```
   <?xml version="1.0"?><DTSConfiguration><DTSConfigurationHeading><DTSConfigurationFileInfo
   GeneratedBy="DOMAIN\UserName" GeneratedFromPackageName="Lesson 5" GeneratedFromPackageID="{F4475E73-
   59E3-478F-8EB2-B10AFA61D3FA}" GeneratedDate="6/10/2012 8:16:50 AM"/></DTSConfigurationHeading>
   <Configuration ConfiguredType="Property"
   Path="\Package.Variables[User::varFolderName].Properties[Value]" ValueType="String"><ConfiguredValue>
   </ConfiguredValue></Configuration></DTSConfiguration>
   ```

   The heading information, **GeneratedBy**, **GeneratedFromPackageID**, and **GeneratedDate** will be different in your file, of course. The element to note is the **Configuration** element. The **Value** property of the variable, `User::varFolderName`, now contains C:\New Sample Data.

3. Save the change, and then close the text editor.

Next Task in Lesson

Step 4: Testing the Lesson 5 Tutorial Package

Lesson 5-4 - Testing the Lesson 5 Tutorial Package

10/1/2018 • 2 minutes to read • Edit Online

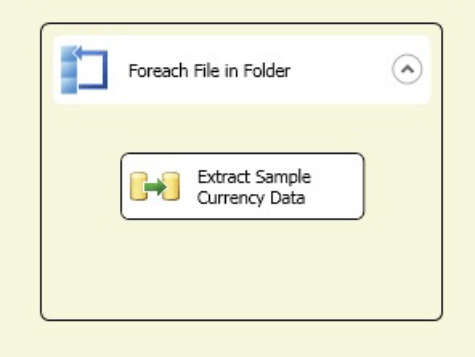
At run time, your package will obtain the value for the **Directory** property from a variable updated at run time, rather than using the original directory name that you specified when you created the package. The value of the variable is populated by the SSISTutorial.dtsConfig file.

To verify that the package updates the Directory property with the new value during run time, simply execute the package. Because only three sample data files were copied to the new directory, the data flow will run only three times, rather than iterate through the 14 files in the original folder.

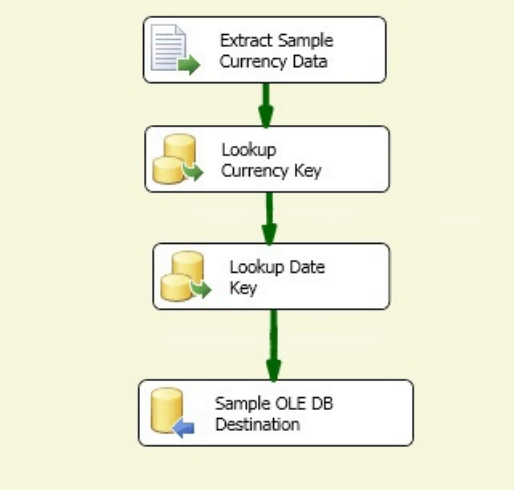
Checking the Package Layout

Before you test the package you should verify that the control and data flows in the Lesson 5 package contains the objects shown in the following diagrams. The control flow should be identical to the control flow in lesson 4. The data flow should be identical to the data flow in lessons 4.

Control Flow



Data Flow



To test the Lesson 5 tutorial package

1. On the **Debug** menu, click **Start Debugging**.
2. After the package has completed running, on the **Debug** menu, and then click **Stop Debugging**.

Next Lesson

Lesson 6: Using Parameters with the Project Deployment Model in SSIS

10/1/2018 • 2 minutes to read • Edit Online

SQL Server 2012 introduces a new deployment model where you can deploy your projects to the Integration Services server. The Integration Services server enables you to manage and run packages, and to configure runtime values for packages.

In this lesson, you will modify the package that you created in Lesson 5: Add SSIS Package Configurations for the Package Deployment Model to use the Project Deployment Model. You replace the configuration value with a parameter to specify the sample data location. You can also copy the completed Lesson 5 package that is included with the tutorial.

Using the Integration Services Project Configuration Wizard, you will convert the project to the Project Deployment Model and use a Parameter rather than a configuration value to set the Directory property. This lesson partially covers the steps you would follow to convert existing SSIS packages to the new Project Deployment Model.

When you run the package again, the Integration Services service uses the parameter to populate the value of the variable, and the variable in turn updates the Directory property. As a result, the package iterates through the files in the new data folder specified by the parameter value rather than the folder that was set in the package configuration file.

> **IMPORTANT**
> This tutorial requires the **AdventureWorksDW2012** sample database. For more information about how to install and deploy **AdventureWorksDW2012**, see Considerations for Installing SQL Server Samples and Sample Databases.

Lesson Tasks

This lesson contains the following tasks:

1. Step 1: Copying the Lesson 5 Package
2. Step 2: Converting the Project to the Project Deployment Model
3. Step 3: Testing the Lesson 6 Package
4. Step 4: Deploying the Lesson 6 Package

Start the Lesson

Step 1: Copying the Lesson 5 Package

Lesson 6-1 - Copying the Lesson 5 Package

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will create a copy of the Lesson 5.dtsx package that you created in Lesson 5. Alternatively, you can add the completed lesson 5 package that is included with the tutorial to the project, and then copy it instead. You will use this new copy throughout the rest of Lesson 6.

To copy the Lesson 5 package

1. If SQL Server Data Tools is not already open, click Start, point to All Programs, point to Microsoft SQL Server 2012, and then click SQL Server Data Tools.
2. On the File menu, click Open, click Project/Solution, select SSIS Tutorial and click Open, and then double-click SSIS Tutorial.sln.
3. In Solution Explorer, right-click Lesson 5.dtsx, and then click Copy.
4. In Solution Explorer, right-click SSIS Packages, and then click Paste.

   By default, the copied package is named Lesson 6.dtsx.
5. In the Solution Explorer, double-click Lesson 6.dtsx to open the package.
6. Right-click anywhere in the background of the Control Flow tab then click Properties.
7. In the Properties window, update the Name property to Lesson 6.
8. Click the box for the ID property, then click the dropdown arrow, and then click .

To add the completed Lesson 5 package

1. Open SQL Server Data Tools and open the SSIS Tutorial project.
2. In Solution Explorer, right-click SSIS Packages, and click Add Existing Package.
3. In the Add Copy of Existing Package dialog box, in Package location, select File system.
4. Click the browse (…) button, Lesson 5.dtsx on your machine, and then click **Open**.

   To download all of the lesson packages for this tutorial, do the following.

   a. Navigate to Integration Services Product Samples

   b. Click the **DOWNLOADS** tab.

   c. Click the SQL2012.Integration_Services.Create_Simple_ETL_Tutorial.Sample.zip file.
5. Copy and paste the Lesson 5 package as described in steps 3-8 in the previous procedure.

   After copying the Lesson 5 package, if you currently have the packages from the previous lessons in your solution, right-click each package from lessons 1-5 and click Exclude From Project. When done you should have only Lesson 6.dtsx in your solution.

Next Task in Lesson

Step 2: Converting the Project to the Project Deployment Model

Lesson 6-2 - Converting the Project to the Project Deployment Model

10/1/2018 • 2 minutes to read • Edit Online

In this task, you will use the Integration Services Project Conversion Wizard to convert the project to the Project Deployment Model.

Converting the Project to the Project Deployment Model

1. On the Project Menu, click Convert to Project Deployment Model.
2. On the Integration Services Project Conversion Wizard Introduction page, review the steps then click Next.
3. On the Select Packages page, in the Packages list, clear all checkboxes except Lesson 6.dtsx then click Next.
4. On the Specify Project Properties page, click Next.
5. On the Update Execute Package Task page click Next.
6. On the Select Configurations page, make sure the Lesson 6.dtsx package is selected in the Configurations list, then click Next.
7. On the Create Parameters page make sure the Lesson 6.dtsx package is selected, and the Scope is set to Package, in the Configuration Properties List, then Click Next.
8. On the Configure Parameters page verify that the values for Name and Value are the same name and value specified in Lesson 5 for the variable and configuration value, then click Next.
9. On the Review page, in the Summary pane, notice that the wizard has used the information from the configuration file to set the Properties to be converted.
10. Click Convert.

    When the conversion completes a message is displayed warning that the changes are not saved until the project is saved in Visual Studio. Click OK on the warning dialog.
11. On the Integration Services Project Conversion Wizard click Close.
12. In SQL Server Data Tools, click the File menu, then click Save to save the converted package.
13. Click the Parameters Tab and verify that the package now contains a parameter for VarFolderName and that the value is the same path specified for the New Sample Data folder from the Lesson 5 configuration file.

Next Task in Lesson

Step 3: Testing the Lesson 6 Package

Lesson 6-3 - Testing the Lesson 6 Package

10/1/2018 • 2 minutes to read • Edit Online

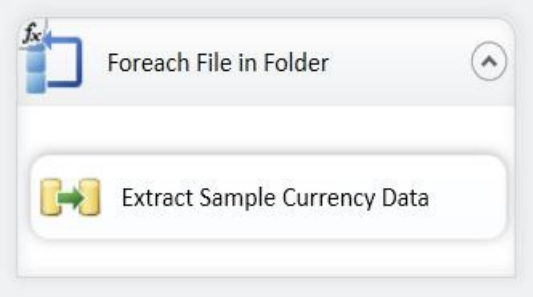
At run time, your package will obtain the value for the Directory property from the VarFolderName parameter.

To verify that the package updates the Directory property with the new value during run time, simply execute the package. Because only three sample data files were copied to the new directory, the data flow will run only three times, rather than iterate through the 14 files in the original folder.

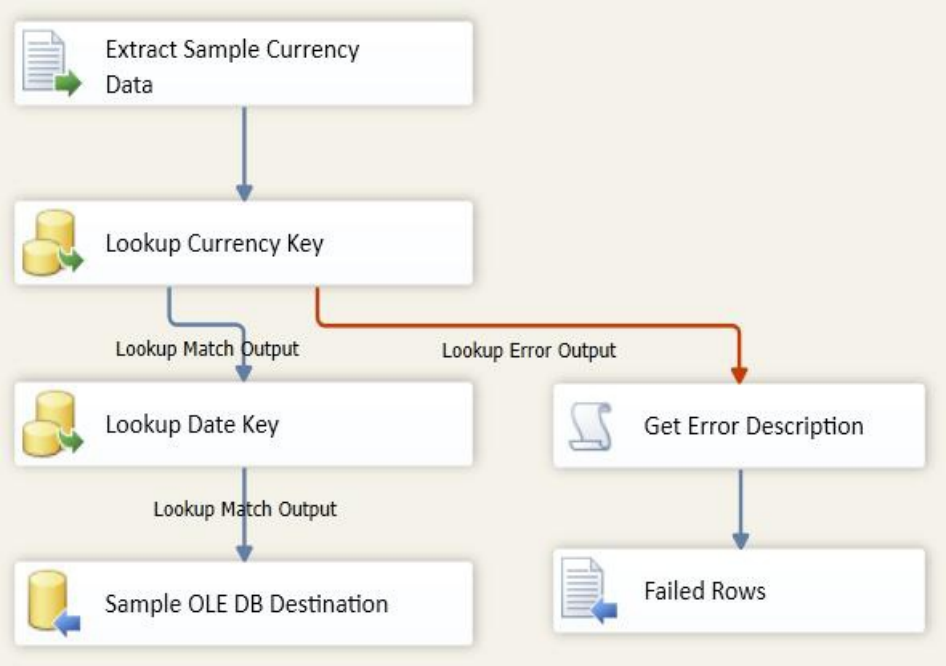
Checking the Package Layout

Before you test the package you should verify that the control and data flows in the Lesson 6 package contains the objects shown in the following diagrams. The control flow should be identical to the control flow in lesson 5. The data flow should be identical to the data flow in lesson 5.

Control Flow



Data Flow



TO test the Lesson 6 tutorial package

1. On the Debug menu, click Start Debugging.
2. After the package has completed running, on the Debug menu, and then click Stop Debugging.

Next Task in Lesson

Step 4: Deploying the Lesson 6 Package

Lesson 6-4 - Deploying the Lesson 6 Package

10/1/2018 • 4 minutes to read • Edit Online

Deploying the package involves adding the package to the SSISDB catalog in Integration Services on an instance of SQL Server. In this lesson you will add the Lesson 6 package to the SSISDB catalog, set the parameter, and execute the package. For this lesson you will use SQL Server Management Studio to add the Lesson 6 package to the SSISDB catalog, and deploy the package. After deploying the package you will modify the parameter to point to a new location then execute the package.

In this lesson you will:

- Add the package to the SSISDB catalog in the SSIS node in SQL Server.
- Deploy the package.
- Set the package parameter value.
- Execute the package in SSMS.

To Locate or add the SSISDB catalog

1. Click Start, point to All Programs, point to Microsoft SQL Server 2012, and then click SQL Management Studio.
2. On the Connect to Server dialog box, verify the default settings, and then click Connect. To connect, the Server name box must contain the name of the computer where SQL Server is installed. If the Database Engine is a named instance, the Server name box should also contain the instance name in the format <computer_name>\<instance_name>.
3. In Object Explorer expand Integration Services Catalogs.
4. If there are no catalogs listed under Integration Services Catalogs then add the SSISDB catalog.
5. To Add the SSISDB catalog, right-click Integration Services Catalogs and click Create Catalog.
6. On the Create Catalog dialog box select Enable CLR Integration.
7. In the Password box, type a new password then type it again in the Retype Password box. Be sure to remember the password you type.
8. Click OK to add the SSISDB catalog.

To add the package to the SSISDB catalog

1. In Object Explorer, right-click SSISDB and click Create Folder.
2. In the Create Folder dialog box type SSIS Tutorial in the Folder name box and click OK.
3. Expand the SSIS Tutorial folder, right-click Projects, and click Import Packages.
4. On the Integration Services Project Conversion Wizard Introduction page click Next.
5. On the Locate Packages page, ensure that File system is selected in the Source list, then click Browse.
6. On the Browse For Folder dialog box, browse to the folder containing the SSIS Tutorial project, then click OK.
7. Click Next.

8. On the Select Packages page you should see all six packages from the SSIS Tutorial. In the Packages list, select Lesson 6.dtsx, then click Next.
9. On the Select Destination page, type SSIS Tutorial Deployment in the Project Name box then click Next.
10. Click Next on each of the remaining wizard pages until you get to the Review page.
11. On the Review page, click Convert.
12. When the conversion completes, click Close.

When you close the Integration Services Project Conversion Wizard, SSIS displays the Integration Services Deployment Wizard. You will use this wizard now to deploy the Lesson 6 package.

1. On the Integration Services Deployment Wizard Introduction page, review the steps for deploying the project, then click Next.
2. On the Select Destination page verify that the server name is the instance of SQL Server containing the SSISDB catalog and that the path shows SSIS Tutorial Deployment, then click Next.
3. On the Review page, review the Summary then click Deploy.
4. When the deployment completes, click Close.
5. In Object Explorer, right-click Integration Services Catalogs and click Refresh.
6. Expand Integration Services Catalogs then expand SSISDB. Continue to Expand the tree under SSIS Tutorial until you have completely expanded the project. You should see Lesson 6.dtsx under the Packages node of the SSIS Tutorial Deployment node.

To verify that the package is complete, right-click Lesson 6.dtsx and click Configure. On the Configure dialog box, select Parameters and verify that there is an entry with Lesson 6.dtsx as the Container, VarFolderName as the Name and the path to New Sample Data as the value, then click Close.

Before continuing create a new sample data folder, name it Sample Data Two, and copy any three of the original sample files into it.

To create and populate a new sample data folder

1. In Windows Explorer, at the root level of your drive (for example, C:\), create a new folder named Sample Data Two.
2. Open the c:\Program Files\Microsoft SQL Server\110\Samples\Integration Services\Tutorial\Creating a Simple ETL Package\Sample Data folder and then copy any three of the sample files from the folder.
3. In the New Sample Data folder, paste the copied files.

To change the package parameter to point to the new sample data

1. In Object Explorer, right click Lesson 6.dtsx and click Configure.
2. On the Configure dialog box, change the parameter value to the path to Sample Data Two. For example C:\Sample Data Two if you placed the new folder in the root folder on the C drive.
3. Click OK to close the Configure dialog box.

To test the Lesson 6 package deployment

1. In Object Explorer, right click Lesson 6.dtsx and click Execute.
2. On the Execute Package dialog box, click OK.
3. On the message dialog box click Yes to open Overview Report.

The Overview report for the package is displayed showing the name of the package and a status summary. The Execution Overview section shows the result from each task in the package and the Parameters Used section shows the names and values of all parameters used in the package execution, including VarFolderName.