5 Tips for a Smooth SSIS Upgrade to SQL Server 2012

Runying Mao Carla Sabotta

Quick Guide

Microsoft

5 Tips for a Smooth SSIS Upgrade to SQL Server 2012

Runying Mao, Carla Sabotta

Summary: Microsoft SQL Server 2012 Integration Services (SSIS) provides significant improvements in both the developer and administration experience. This article provides tips that can help to make the upgrade to Microsoft SQL Server 2012 Integration Services successful. The tips address editing package configurations and specifically connection strings, converting configurations to parameters, converting packages to the project deployment model, updating Execute Package tasks to use project references and parameterizing the PackageName property.

Category: Quick Guide
Applies to: SQL Server 2012
Source: White paper (link to source content)
E-book publication date: November 2012

Contents

Introduction

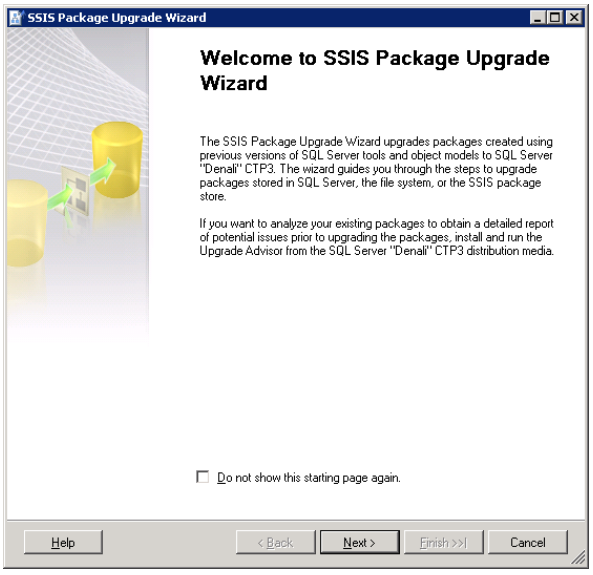
Microsoft SQL Server 2012 Integration Services (SSIS) provides significant improvements in both the developer and administration experience. New SSIS features have been introduced in order to improve developer productivity, and simplify the deployment, configuration, management and troubleshooting of SSIS packages.

SQL Server 2012 Integration Services introducesthe project as a self-contained, deployment unit. Common values can be shared among packages in the same project through project parameters and project connection managers. ETL developers caneasily reference child packages that are inside the project.

Solutions that were created in earlier versions of SSIS (pre-SQL Server 2012) will be supported in SQL Server 2012. When you upgrade the solutions, you can take advantage of the new SQL Server 2012 features. Although in SQL Server 2012 SSIS offers wizards for upgrading most solution components, there will be a few settings that you'll need to change manually.

Here are a few tips that can help to make the upgrade successful.

TIP #1: Edit Package Configuration and Data Source after upgrading



The first step to upgrade an SSIS solution is to run the SSIS Package Upgrade Wizard. The SSIS Package Upgrade Wizard makes appropriate changes to package properties and upgrades the package format.

The wizard launches when you open a pre-SQL Server 2012 package in the SQL Server Data Tools for the first time. SQL Server Data Tools replaces (BIDs). The wizard can also be launched manually by

running**SSISUpgrade.exe,** which is located under **%ProgramFiles%\Microsoft SQL Server\110\DTS\Binn**.

It is critical to note that the SSIS Package Upgrade Wizard does not upgrade settings such as connection strings that are defined in the package configurations. After a package upgrade, you may need to make some manual changes to the package configuration to run the upgraded package successfully.

For example, you have an SSIS 2005 package. The package uses an OLE DB connection manager to connect to the AdventureWorks database in a local SQL Server 2005 instance. The package also uses an XML package configuration file to dynamically configure the ConnectionString property of the OLE DB connection manager. The following shows the contents of the XML package configuration file.

Sample: XML Package Configuration File for an SSIS 2005 package

```
<?xml version="1.0"?>

<DTSConfiguration>

  <Configuration ConfiguredType="Property" Path="\Package.Connections[AdventureWorks].ConnectionString"
ValueType="String">

    <ConfiguredValue>Data Source=(local);Initial Catalog=AdventureWorks;Provider=SQLNCLI.1;Integrated
Security=SSPI;</ConfiguredValue>

  </Configuration>

</DTSConfiguration>
```

You have set up a machine with a standalone SQL Server 2012 installation. You move the SSIS 2005 package to the machine and run the SSIS Package Upgrade Wizard to upgrade the package to SQL Server 2012. When the wizard finishes, you need to manually change the provider name from SQLNCLI.1 to SQLNCLI11.1 in the XML package configuration file to run the upgraded package successfully. The wizard does not update package configuration files.

If you don't update the provider name in the configuration file, the file configures the OLE DB connection manager to use the SQLNCLI.1 provider that is the SQL Server 2005 Native Client Library. SQLNCLI11.1 is the SQL Server 2012 Native Client Library. Because the SQL Server 2005 Native Client Library is not included in SQL Server 2012, the following error message will appear when you open or execute the upgraded package on the machine where SQL Server 2012is installed:

> **The requested OLE DB provider SQLNCLI.1 is not registered. If the 32-bit driver is not installed, run the package in 64-bit mode. Error code: 0x00000000. An OLE DB record is available. Source: "Microsoft OLE DB Service Components" Hresult: 0x80040154 Description: "Class not registered".**

So, if your pre-SQL Server 2012 package uses any kind of package configurations, it is important to remember that you may need to manually update the content of the package configurations after you upgrade the package to SQL Server 2012. This applies to the different types of configurations, such as XML configuration files.

Connection strings that require updates and are stored in data source files or set by expressions, need to be updated manually.

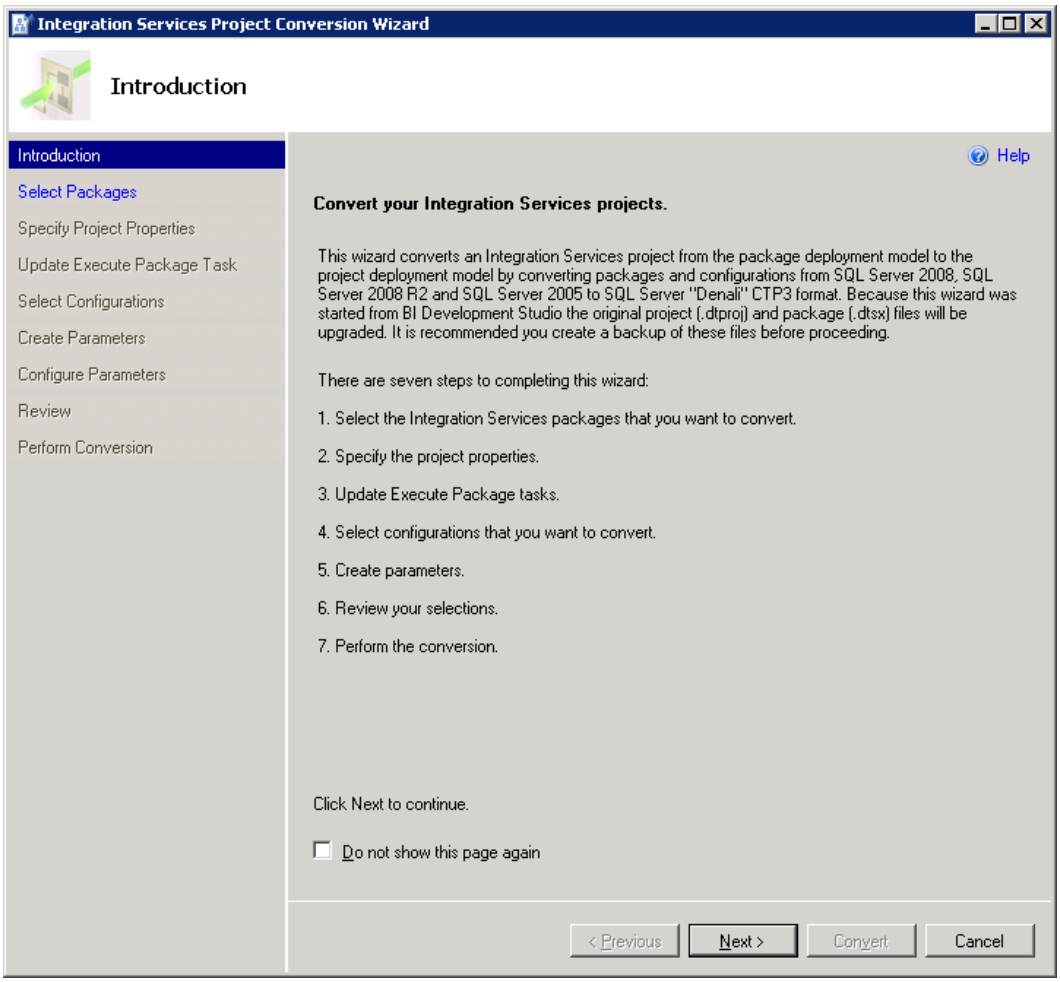
TIP #2: Convert to project deployment model using Project Conversion Wizard

SQL Server 2012 SSIS supports two deployment models: the package deployment model and the project deployment model. The package deployment model was available in previous releases of SSIS and is the default deployment model for upgraded packages. In this model, the unit of deployment is the package. The project deployment model is new in SQL Server 2012 and provides additional package deployment and management features such as parameters and the Integration Services catalog. The unit of deployment is the project.

Please read Project Deployment Overview in SQL Server "Denali" CTP1 - SSIS(http://social.technet.microsoft.com/wiki/contents/articles/project-deployment-overview-in-sql-server-quot-denali-quot-ctp1-ssis.aspx) for a detailed walk through as well as comparison between these two deployment models.

Read Projects in SQL Server “Denali” CTP1 - SSIS(http://social.technet.microsoft.com/wiki/contents/articles/projects-in-sql-server-denali-ctp1-ssis.aspx) for a thorough explanation of the new project concept.

To convert a package to the project deployment, right click the project in **Solution Explorer** and then click **Convert to Project Deployment Model**. The Project Conversion Wizard launches and walks you through the conversion process.



TIP #3: Update Execute Package Task to use project reference and use parameter to pass data from parent package to child package

If an SSIS package contains an Execute Package Task, the Project Conversion Wizard prompts you to update the task to use the project reference.

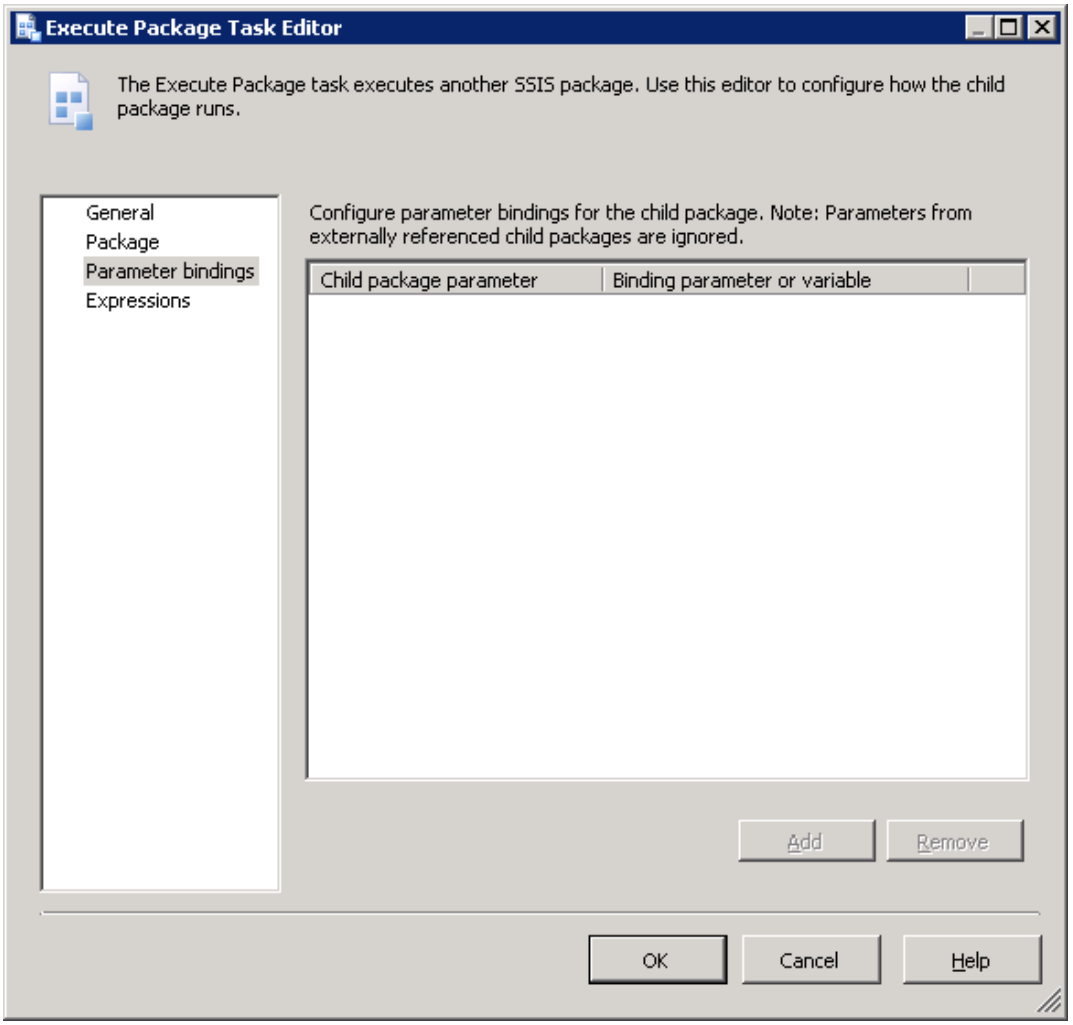
For example, your SSIS project contains several packages. Inside the project, one package (typically called the parent package) runs another package (typically called the child package) by using an Execute Package Task. In Pre-SQL Server 2012 releases of SSIS, the parent package references the child package by using a File connection manager. At deployment, you need to remember to update the File connection manager to ensure that it points to the new location of the child package.

In SQL Server 2012 Integration Services you can configure the parent package to reference the child package by name when the child package is included in the same project as the parent package. Using this project reference makes the deployment experience much smoother. You don't need to remember to update the reference between the parent package and the child package at deployment. For a

thorough explanation of the project reference in the Execute Package Task, please see [Changes to the Execute Package Task](http://blogs.msdn.com/b/mattm/archive/2011/07/18/changes-to-the-execute-package-task.aspx).

In previous releases of SSIS, you pass data from the parent package to the child package by creating a package configuration that uses the parent variable configuration type. This enables a child package that is run from a parent package to access a variable in the parent.

It is recommended that you configure the Execute Package Task to use parameter binding to pass data from the parent package to the child package. Parameters make this task easier. For example, you want a parent package to dynamically determine the number of days in a current month and have the child package perform a task for that number of times. You can create a variable in the parent package that represents the number of days and create a parameter in the child package. Then in the Execute Package Task, you bind the parameter in the child package to the variable in the parent package.



Please read Parameters in SQL Server "Denali" CTP1 - SSIS (http://social.technet.microsoft.com/wiki/contents/articles/parameters-in-sql-server-denali-ctp1-ssis.aspx) for a description of parameters and the numerous benefits they offer.

TIP #4: Parameterize PackageName property of Execute Package Task to dynamically configure which child package to run at execution time

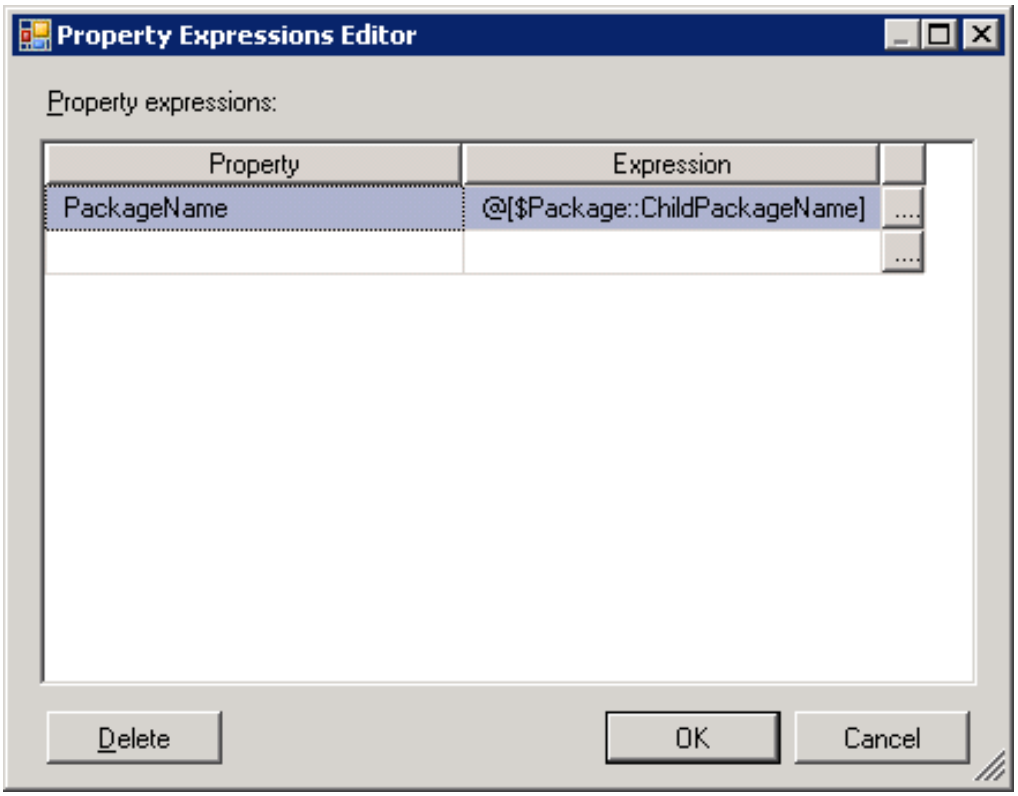
Suppose your SSIS 2008 package has an Execute Package Task, and the package uses a File connection manager to connect to a child package. You dynamically assign which child package the Execute Package Task runs by configuring the connection string property of the File connection manager.

The following is the content of the XML package configuration file used by your SSIS 2008 package.

Sample: XML Package Configuration File for an SSIS 2008 package

```
<?xml version="1.0"?>

<DTSConfiguration>

 <Configuration ConfiguredType="Property" Path="\Package.Connections[Child.dtsx].Properties[ConnectionString]"
ValueType="String">

  <ConfiguredValue>E:\Integration Services Project1\Integration Services
Project1\Child.dtsx</ConfiguredValue>

 </Configuration>

</DTSConfiguration>
```

When the Project Conversion Wizard converts the package to the project deployment model and updates the Execute Package Task to use the project reference, the File connection manager that was used to connect to the child package is no longer used by the Execute Package Task. To continue to dynamically determine which child package the task runs, you create a parameter and map that parameter to the PackageName property of the Execute Package Task as shown in the following image.



TIP #5: Convert package configuration to parameter when possible

Parameters are new to SQL Server 2012 Integration Services and are the replacement for package configurations. You use parameters to assign values to package properties, whether at design time or run time. The values are pushed to a package when it is executed rather than having the package pull values from the package configurations.

The Project Conversion Wizard prompts you to optionally convert package configurations to parameters. It is possible that you might choose to keep a package configuration as an intermediate step of upgrading to SQL Server 2012. When your package has both configuration values and parameter values, it is important to understand the order in which these values are applied. Package configuration values will be applied first. If there are also parameter values for the same properties, these values will be applied next and will overwrite the package configuration values.

Conclusion

Microsoft SQL Server 2012 Integration Services (SSIS) offers features that greatly enhance the development and administrative experience. These tips could help users ensure successful upgrades of their current solutions to SQL Server 2012 so that they can take advantage of SQL Server 2012's new features. For more information about SQL Server 2012 Integration Services and what's new, please refer to What's New (Integration Services) (http://msdn.microsoft.com/en-us/library/bb522534(v=SQL.110).aspx).

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

- Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
- Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

Send feedback.