Executing SSIS Packages

30 August 2006
by Nigel Rivett

Executing all SSIS packages in a folder: three methods

A common requirement is to execute several SSIS packages as part of one logical process. This is especially true when one is transferring existing data from an external system.

If the indivisual packages are placed in a folder and named in such a way that a batch can be executed via a wildcard filemask, then they can be altered without changing any code, and can be used for different processes by changing the filemask. This article presents three different methods of executing all packages in a folder:

1. SSIS control package
2. Stored procedure
3. DOS batch file

The methods are of differing complexity and flexibility, the SSIS package being by far the most difficult to code, and the DOS command being by far the simplest but also the least flexible.

The folder that contains the packages is d:\TestPackages\.

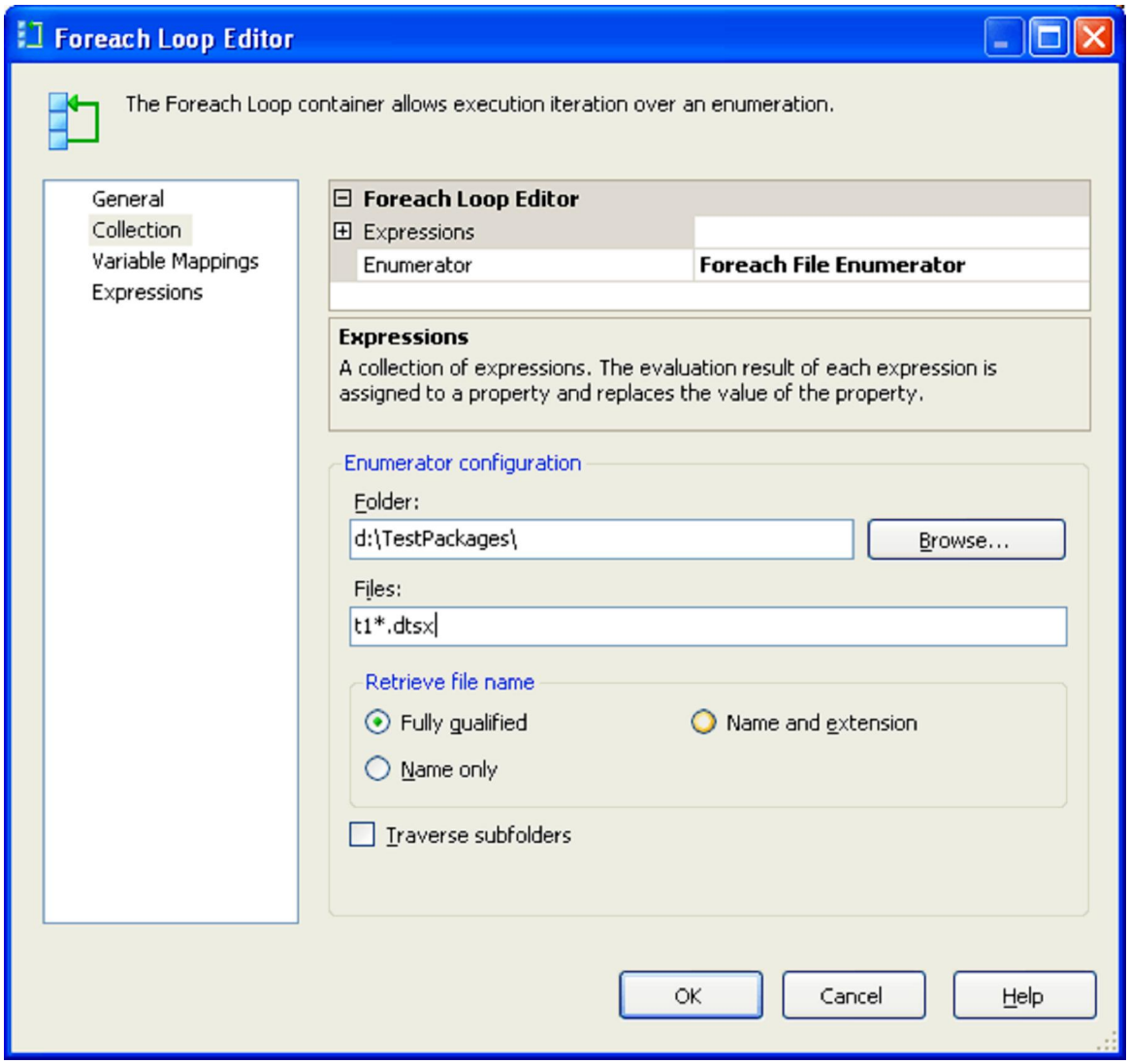
All the packages to be run have names of the format t1…….dtsx.

SSIS control package

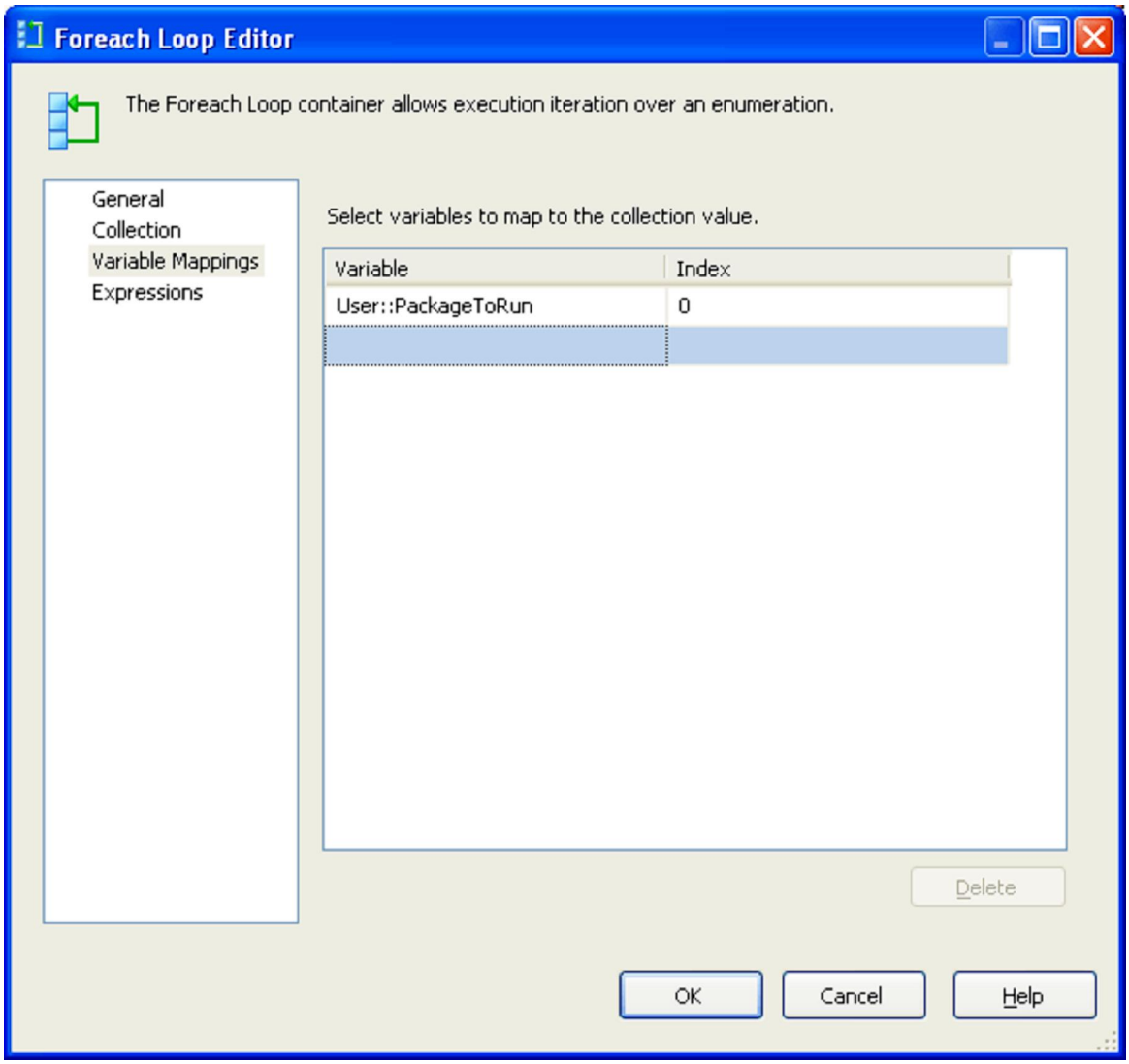
For this we create a package that uses a for each loop task to call an execute package task for all packages in the folder.

The first task is to create an SSIS package with a Foreach Loop container that will loop round the packages in the folder, setting a variable called "PackageToRun" to the file name for each package.

1. Load Business Intelligence Development Studio and start a SSIS project.
2. Create a new package.
3. Add a Foreach Loop container to the package.
4. Right-click on the **Foreach Loop** container and select **Edit**.
5. Click on **Collection**.
6. Set the Enumerator to **Foreach File Enumerator**.
7. In the Enumerator configuration:
    a. Set **Folder** to "d:\TestPackages\"
    b. Set **Files** to "t1*.dtsx"
    c. Under **Retrieve file name** select **Fully qualified**.
8. Click **OK**.

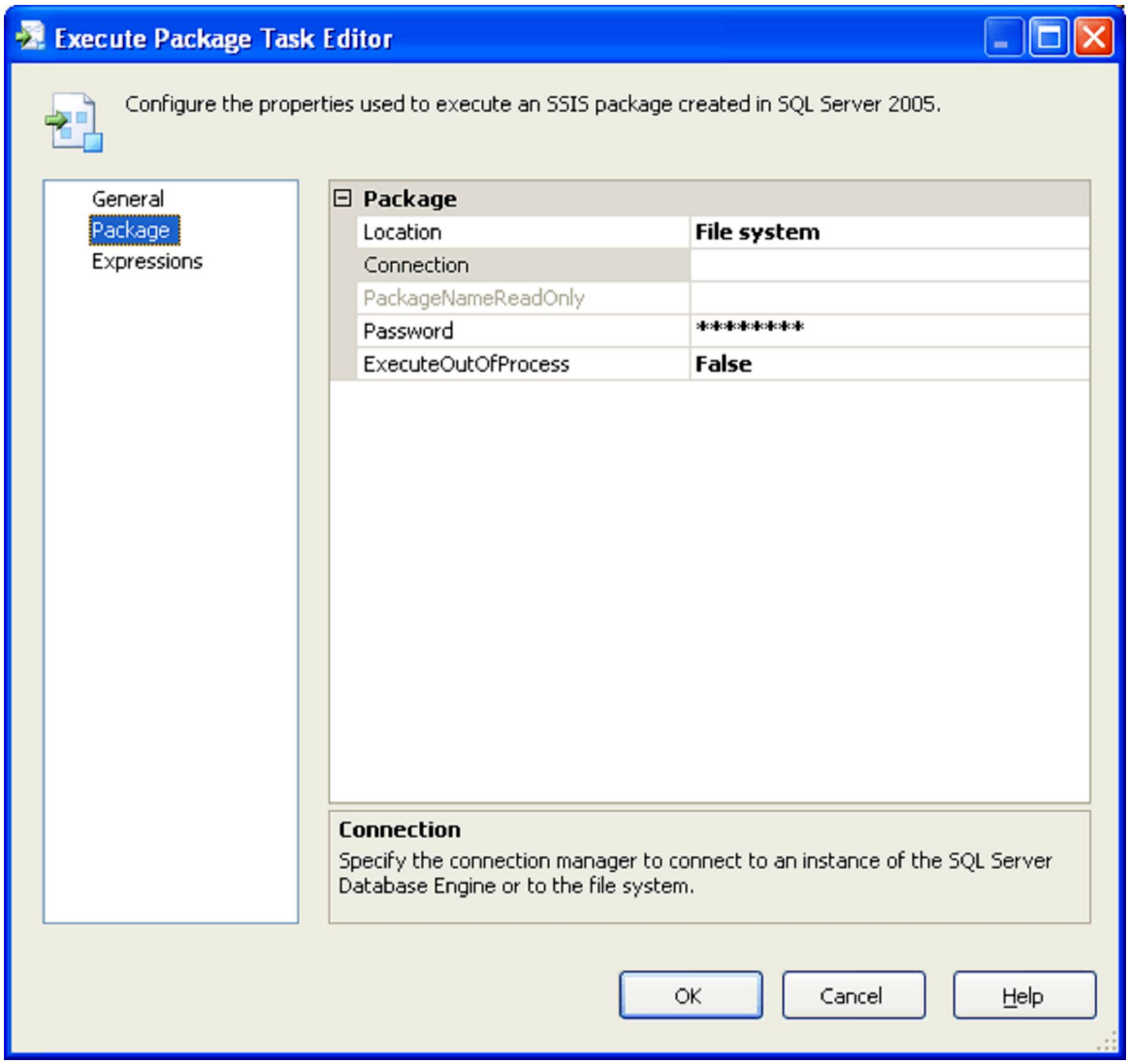


9. Click on **ariale Mappings** .
10. Click on the **ariale** drop-down list and select **e ariale** .
11. Set Name to PackageToRun.
12. Click **OK**.

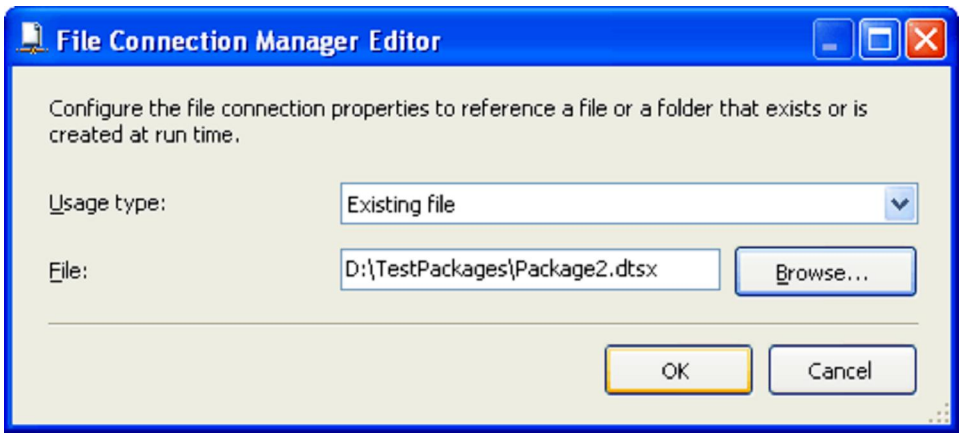


Next, we need to add the **Execute Package** task to the Foreach Loop container so that this task will be executed for each package that we wish to run. e then set the variable value to be the name of the package to be executed by the **ExecutePackage** task.

13. Drag an **Execute Package** task into the Foreach Loop container.
14. Right-click on the **Execute Package** task and select **Edit**.
15. Select **Package**.
16. Set Location to File system



17. Click on the **Connection** drop-down list and select New connection…
18. Set the File to an existing package.



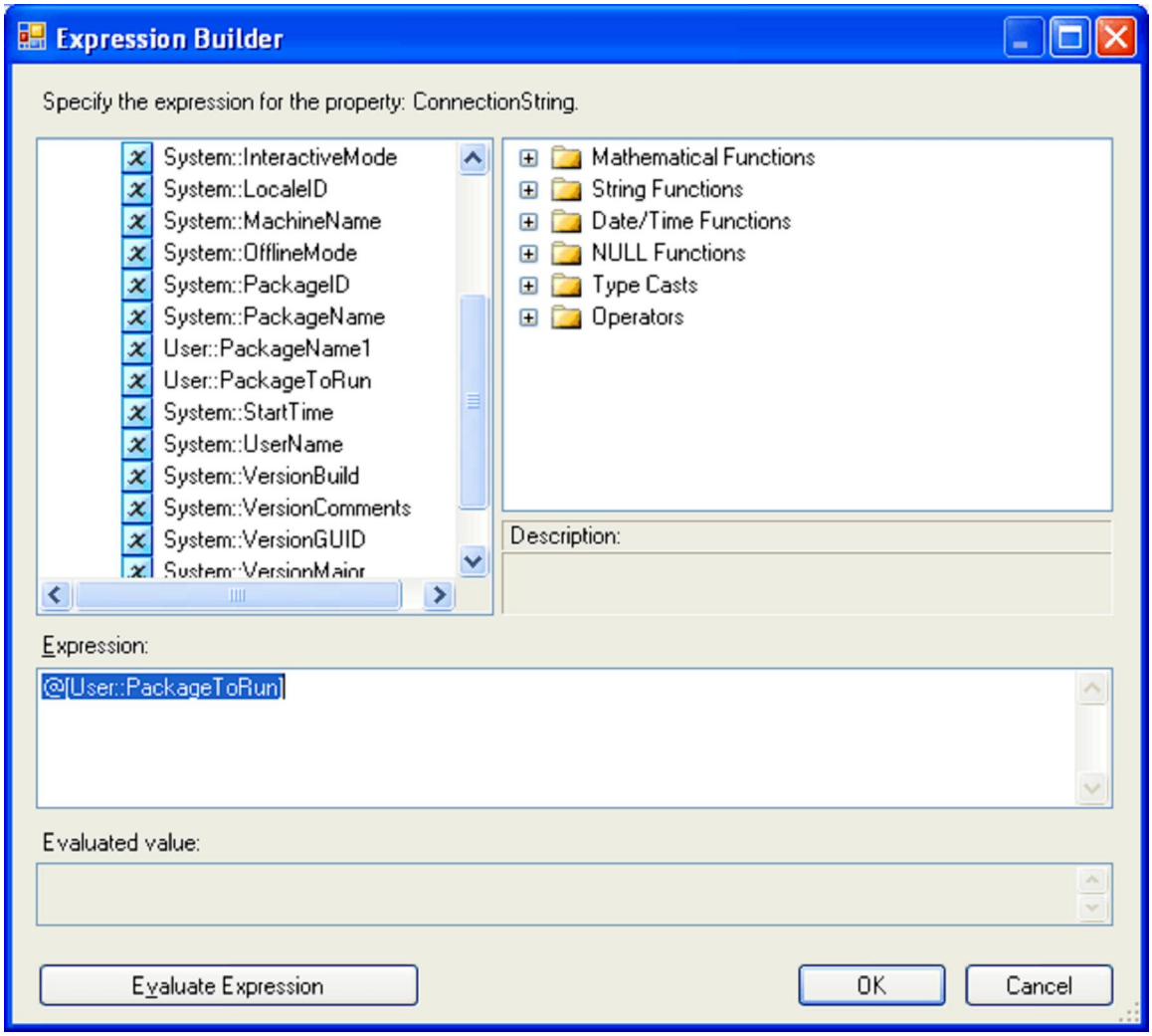
1. Click **OK** to save the connection.
20. Click **OK** to complete the Execute Package task configuration.

Finally, we configure the connection to use the variable package name:

21. Right-click on the connection and select **Properties**.
22. In the Properties window change the name to PackageToRunConnection.

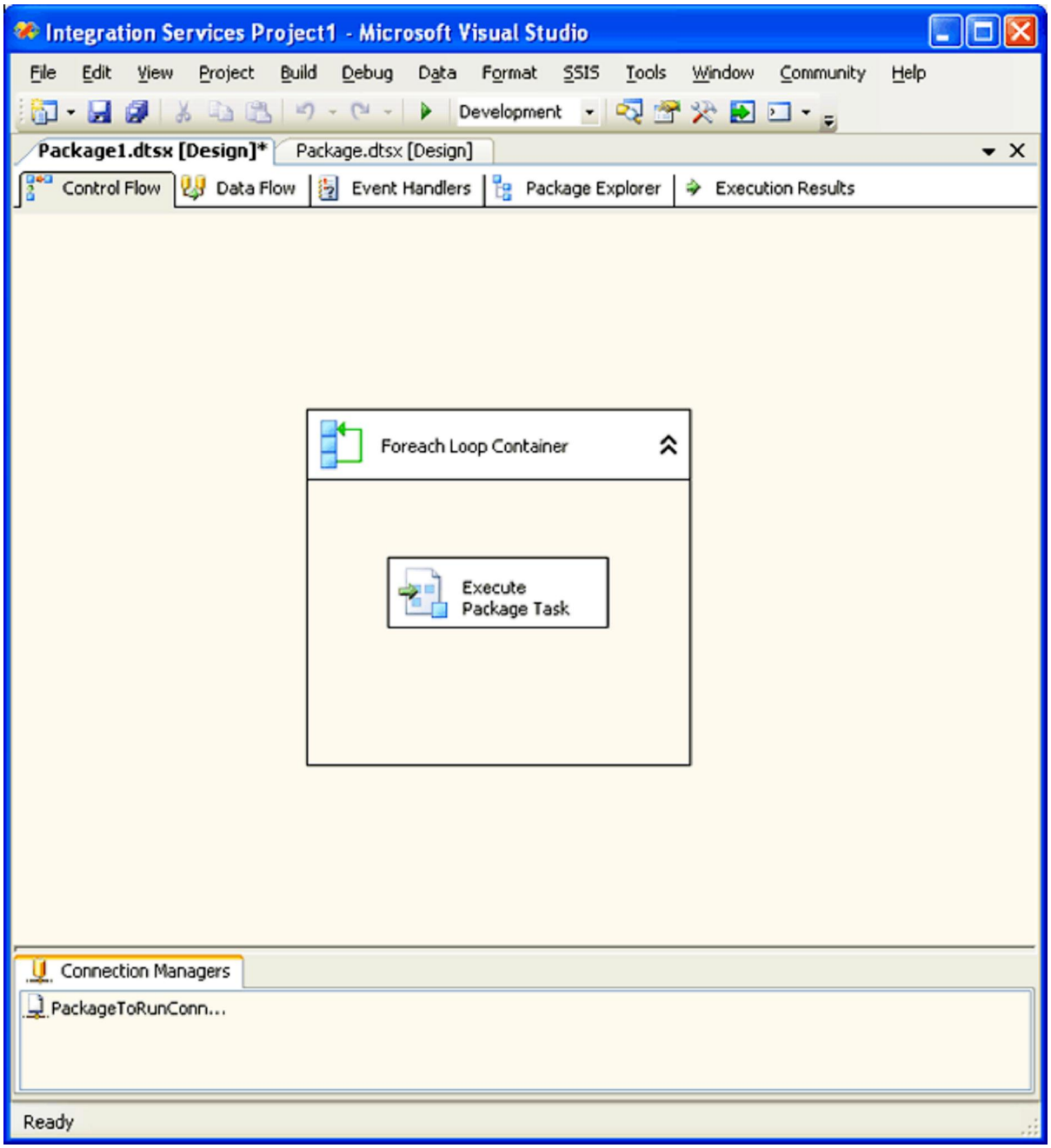
***Note**: this will automatically update the connection name in the Execute Package Task.*

23. Select Expressions and add a new expression.
24. In the property drop-down list select **ConnectionString**.
25. Click on the **Expression Editor** button.
26. From the ariables tree drag User::PackageToRun into the Expression window.



27. Click **OK** twice, to save the expression.

The package should now look like this:


Integration Services Project1 - Microsoft Visual Studio
File Edit View Project Build Debug Data Format SSIS Tools Window Community Help
Development
Package1.dtsx [Design]*
Package.dtsx [Design]
Control Flow
Data Flow
Event Handlers
Package Explorer
Execution Results
Foreach Loop Container
Execute Package Task
Connection Managers
PackageToRunConn...
Ready

```
Create table #dir (Filename varchar(1000))
Insert #dir
Exec master..xp_cmdshell 'dir /B d:\TestPackages\ t1*.dtsx'
delete #dir where Filename is null or Filename like '%not found%'
```

```
Declare @Filename varchar(1000)
Select @Filename = ''
While @Filename < (select max(Filename) from #dir)
Begin
   Select @Filename = min(Filename) from #dir where Filename > @Filename
end
```

```
Declare @cmd varchar(1000)
select @cmd = 'dtexec /F " d:\TestPackages\' + @FileName + '''
exec master..xp_cmdshell @cmd
```

```
Declare @Filename varchar(1000)
Declare @cmd varchar(1000)
Create table #dir (Filename varchar(1000))
Insert #dir
Exec master..xp_cmdshell 'dir /B d:\TestPackages\t1*.dtsx'
delete #dir where Filename is null or Filename like '%not found%'
Select @Filename = ''
While @Filename < (select max(Filename) from #dir)
Begin
   Select @Filename =  min(Filename) from #dir where Filename > @Filename
   select  @cmd =  'dtexec /F "d:\TestPackages\' + @Filename + '"'
   exec master..xp_cmdshell @cmd
end
drop table #dir
```

```
Create procedure [dbo].[s_ProcessAllFilesInDir]
@FilePath varchar(1000) ,
@FileMaskvarchar(100) ,
@ProcSp varchar(128)
as
   set nocount on

declare @File varchar(128) ,
      @MaxFile varchar(128) ,
      @cmd     varchar(2000)

   create table #Dir (s varchar(8000))
```

```
   select @cmd = 'dir /B ' + @FilePath + @FileMask
   insert #Dir exec master..xp_cmdshell @cmd

   delete #Dir where s is null or s like '%not found%'

   select @File = '', @MaxFile = max(s) from #Dir
   while @File < @MaxFile
   begin
      select @File = min(s) from #Dir where s > @File

      select @cmd = @ProcSp + ' ''' + @FilePath + ''' , ''' + @File + ''''

      exec (@cmd)

   end
   drop table #Dir
go
```

```
Create procedure [dbo].[s_ExecutePackage]
@FilePath   varchar(1000) ,
@Filename   varchar(128)
as
Declare @cmd varchar(1000)
   select @cmd =  'dtexec /F "' + @FilePath + @Filename + '"'
   exec master..xp_cmdshell @cmd
go
```

```
Exec [dbo].[s_ProcessAllFilesInDir]
   @FilePath = 'd:\TestPackages\' ,
   @FileMask = 't1*.dtsx' ,
   @ProcSp = '[dbo].[s_ExecutePackage]'
```

```
for %1 in (d:\TestPackages\t1*.dtsx) do dtexec /F "%1"
```