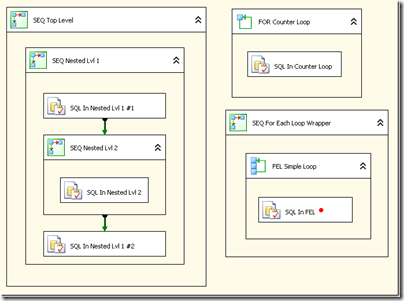
**[Exploring packages in code](http://www.sqlis.com/post/Exploring-packages-in-code.aspx)**

By [Darren Green](http://www.sqlis.com/author/Darren%20Green.aspx)17 jul 2009 16:07

In my previous post [Searching for tasks with code](http://www.sqlis.com/post/Searching-for-tasks.aspx) you can see how to explore the control flow side of packages, drilling down through containers, task, and event handlers, but it didn’t cover the data flow. I recently saw a post on the MSDN forum asking how to edit an existing package programmatically, and the sticking point was how to find the the data flow and the components inside.

This post builds on some of the previous code and shows how you can explore all objects inside a package. I took the sample [Task Search](http://www.sqlis.com/file.axd?file=WindowsLiveWriter/556b7e429c30/2876C1ED/TaskSearch.zip) application I’d written previously, and came up with a totally pointless little console application that just walks through the package and writes out the basic type and name of every object it finds, starting with the package itself e.g. *Package – MyPackage* .

The sample package we used last time showed nested objects as well an event handler; a *OnPreExecute* event tucked away on the task *SQL In FEL*.

[](http://www.sqlis.com/image.axd?picture=WindowsLiveWriter/Exploringpackagesincode/1F0980BE/EventsAndContainersWithExecSQLForSea1.png)

The output of this sample tool would look like this:

PackageObjects v1.0.0.0 (1.0.0.26627)   
Copyright (C) 2009 Konesans Ltd

Processing File - Z:\Users\Darren Green\Documents\Visual Studio 2005\Projects\SSISTestProject\EventsAndContainersWithExe   
cSQLForSearch.dtsx

Package - EventsAndContainersWithExecSQLForSearch   
For Loop - FOR Counter Loop   
Task - SQL In Counter Loop   
Sequence Container - SEQ For Each Loop Wrapper   
For Each Loop - FEL Simple Loop   
Task - SQL In FEL   
Task - SQL On Pre Execute for FEL SQL Task   
Sequence Container - SEQ Top Level   
Sequence Container - SEQ Nested Lvl 1   
Sequence Container - SEQ Nested Lvl 2   
Task - SQL In Nested Lvl 2   
Task - SQL In Nested Lvl 1 #1   
Task - SQL In Nested Lvl 1 #2   
Connection Manager – LocalHost

The code is very similar to what we had previously, but there are a couple of extra bits to deal with connections and to look more closely at a task and see if it is a Data Flow task.

For connections your just examine the package's Connections collection as shown in the abridged snippets below. First you can see the call to the *ProcessConnections* method, followed by the method itself.

// Load the package file

Application application = new Application();

using (Package package = application.LoadPackage(filename, null))

{

// Write out the package name

Console.WriteLine("Package - {0}", package.Name);

... More ...

// Look and the connections

ProcessConnections(package.Connections);

}

private static void ProcessConnections(Connections connections)

{

foreach (ConnectionManager connectionManager in connections)

{

Console.WriteLine("Connection Manager - {0}", connectionManager.Name);

}

}

What we didn’t see in the sample output above was anything to do with the Data Flow, but rest assured the code now handles it too. The following snippet shows how each task is examined to see if it is a Data Flow task, and if so we can then loop through all of the components inside the data flow.

private static void ProcessTaskHost(TaskHost taskHost)

{

if (taskHost == null)

{

return;

}

Console.WriteLine("Task - {0}", taskHost.Name);

// Check if the task is a Data Flow task

MainPipe pipeline = taskHost.InnerObject as MainPipe;

if (pipeline != null)

{

ProcessPipeline(pipeline);

}

}

private static void ProcessPipeline(MainPipe pipeline)

{

foreach (IDTSComponentMetaData90 componentMetadata in pipeline.ComponentMetaDataCollection)

{

Console.WriteLine("Pipeline Component - {0}", componentMetadata.Name);

// If you wish to make changes to the component then you should really use the managed wrapper.

// CManagedComponentWrapper wrapper = componentMetadata.Instantiate();

// wrapper.SetComponentProperty("PropertyName", "Value");

}

}

Hopefully you can see how we get a reference to the Data Flow task, and then use the ComponentMetaDataCollection to find out what components we have inside the pipeline. If you wanted to know more about the component you could look at the ObjectType or ComponentClassID properties. After that it gets a bit harder and you should get a reference to the wrapper object as the comment suggest and start using the properties, just like you would in the create packages samples, see our [Code Development](http://www.sqlis.com/category/Code-Development.aspx) category for some for these examples.

**Download**

Sample code project [PackageObjects.zip](http://www.sqlis.com/file.axd?file=WindowsLiveWriter/Exploringpackagesincode/338ABC3D/PackageObjects.zip) (5KB)