SQL Server 2008 Books Online (October 2008)

Raising and Defining Events in a Data Flow Component

Component developers can raise a subset of the events defined in the IDTSComponentEvents [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.runtime.idtscomponentevents.aspx] interface by calling the methods exposed on the ComponentMetaData [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.pipelinecomponent.componentmetadata.aspx] property. You can also define custom events by using the EventInfos [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.pipelinecomponent.eventinfos.aspx] collection, and raise them during execution by using the FireCustomEvent [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.wrapper.idtscomponentmetadata100.firecustomevent.aspx] method. This section describes how to create and raise an event, and provides guidelines on when you should raise events at design time.

Raising Events

Components raise events by using the **Fire<X>** methods of the IDTSComponentMetaData100 [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.wrapper.idtscomponentmetadata100.aspx] interface. You can raise events during component design and execution. Typically, during component design, the FireError [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.wrapper.idtscomponentmetadata100.fireerror.aspx] and FireWarning [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.wrapper.idtscomponentmetadata100.firewarning.aspx] methods are called during validation. These events display messages in the **Error List** pane of Business Intelligence Development Studio and provide feedback to users of the component when a component is incorrectly configured.

Components can also raise events at any point during execution. Events allow component developers to provide feedback to users of the component as it executes. Calling the **FireError** method during execution is likely to fail the package.

Defining and Raising Custom Events

Defining a Custom Event

Custom events are created by calling the Add [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.runtime.wrapper.idtseventinfos100.add.aspx] method of the **EventInfos** collection. Th collection is set by the data flow task and provided as a property to the component developer through the PipelineComponent [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.pipelinecomponent.aspx] base class. This class contains custom events defined by the data flow task and custom events defined by the component during the RegisterEvents [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.pipeline.pipelinecomponent.registerevents.aspx] method.

The custom events of a component are not persisted in the package XML. Therefore, the **RegisterEvents** method is called during both design and execution to allow the component to define the events it raises.

The *allowEventHandlers* parameter of the **Add** method specifies whether the component allows DtsEventHandler [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.runtime.dtseventhandler.aspx] objects to be created for the event. Note that DtsEventHandlers [http://msdn.microsoft.com/en-us/library/microsoft.sqlserver.dts.runtime.dtseventhandlers.aspx] are synchronous. Therefore the component does not resume execution until an **DtsEventHandler** attached to the custom event has finished executing. If the *allowEventHandlers* parameter is **true**, each parameter of the event is automatically made available to any **DtsEventHandler** objects through variables that are created and populated automatically by the SQL Server Integration Services runtime.

Raising a Custom Event

Components raise custom events by calling the **FireCustomEvent** method, and providing the name, text, and parameters of the event. If the *allowEventHandlers* parameter is **true**, any **DtsEventHandlers** that are created for the custom event are executed by the SSIS run-time engine.

Custom Event Sample

The following code example shows a component that defines a custom event during the **RegisterEvents** method, and then raises the event at run time by calling the **FireCustomEvent** method.

C# Copy Coc

```
public override void RegisterEvents()
{
   string [] parameterNames = new string[2]{"RowCount", "StartTime"};
   ushort [] parameterTypes = new ushort[2]{ DtsConvert.VarTypeFromTypeCode(TypeCode.Int32), DtsConvert.VarTypeFromTypeCode(TypeCode.DateTime)};
   string [] parameterDescriptions = new string[2]{"The number of rows to sort.", "The start time of the Sort operation."};
   EventInfos.Add("StartingSort","Fires when the component begins sorting the rows.",false,ref parameterNames, ref paramterTypes, ref parameterDescriptions);
}
public override void ProcessInput(int inputID, PipelineBuffer buffer)
{
   while (buffer.NextRow())
   {
     // Process buffer rows.
   }

   IDTSEventInfo100 eventInfo = EventInfos["StartingSort"];
   object []arguments = new object[2]{buffer.RowCount, DateTime.Now };
   ComponentMetaData.FireCustomEvent("StartingSort", "Beginning sort operation.", ref arguments, ComponentMetaData.Name, ref FireSortEventAgain);
}
```

Visual Basic Copy Coc

```
Public  Overrides Sub RegisterEvents()
  Dim parameterNames As String() = New String(2) {"RowCount", "StartTime"}
  Dim parameterTypes As System.UInt16() = New System.UInt16(2) {DtsConvert.VarTypeFromTypeCode(TypeCode.Int32), DtsConvert.VarTypeFromTypeCode(TypeCode.DateTime)}
  Dim parameterDescriptions As String() = New String(2) {"The number of rows to sort.", "The start time of the Sort operation."}
  EventInfos.Add("StartingSort", "Fires when the component begins sorting the rows.", False, parameterNames, paramterTypes, parameterDescriptions)
End Sub

Public  Overrides Sub ProcessInput(ByVal inputID As Integer, ByVal buffer As PipelineBuffer)
  While buffer.NextRow
  End While
  Dim eventInfo As IDTSEventInfo100 = EventInfos("StartingSort")
  Dim arguments As Object() = New Object(2) {buffer.RowCount, DateTime.Now}
  ComponentMetaData.FireCustomEvent("StartingSort", _
    "Beginning sort operation.", arguments, _
    ComponentMetaData.Name, FireSortEventAgain)
End Sub
```

See Also

Other Resources

Integration Services Event Handlers [http://msdn.microsoft.com/en-us/library/ms140223.aspx]
Creating Package Event Handlers [http://msdn.microsoft.com/en-us/library/ms140011.aspx]

Help and Information

Getting SQL Server 2008 Assistance [http://msdn.microsoft.com/en-us/library/ms166016.aspx]

Tags:

Community Content