|  |  |
| --- | --- |
|  |  |
|  |  |
| #Region "Help: Introduction to the script task"  'The Script Task allows you to perform virtually any operation that can be accomplished in  'a .Net application within the context of an Integration Services control flow.  'Expand the other regions which have "Help" prefixes for examples of specific ways to use  'Integration Services features within this script task.  #End Region  #Region "Imports"  Imports System  Imports System.Data  Imports System.Math  Imports Microsoft.SqlServer.Dts.Runtime  #End Region  'ScriptMain is the entry point class of the script. Do not change the name, attributes,  'or parent of this class.  <Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute()> \_  <System.CLSCompliantAttribute(False)> \_  Partial Public Class ScriptMain  Inherits Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase  #Region "Help: Using Integration Services variables and parameters in a script"  'To use a variable in this script, first ensure that the variable has been added to  'either the list contained in the ReadOnlyVariables property or the list contained in  'the ReadWriteVariables property of this script task, according to whether or not your  'code needs to write to the variable. To add the variable, save this script, close this instance of  'Visual Studio, and update the ReadOnlyVariables and  'ReadWriteVariables properties in the Script Transformation Editor window.  'To use a parameter in this script, follow the same steps. Parameters are always read-only.  'Example of reading from a variable:  ' startTime = Dts.Variables("System::StartTime").Value  'Example of writing to a variable:  ' Dts.Variables("User::myStringVariable").Value = "new value"  'Example of reading from a package parameter:  ' batchId = Dts.Variables("$Package::batchId").Value  'Example of reading from a project parameter:  ' batchId = Dts.Variables("$Project::batchId").Value  'Example of reading from a sensitive project parameter:  ' batchId = Dts.Variables("$Project::batchId").GetSensitiveValue()  #End Region  #Region "Help: Firing Integration Services events from a script"  'This script task can fire events for logging purposes.  'Example of firing an error event:  ' Dts.Events.FireError(18, "Process Values", "Bad value", "", 0)  'Example of firing an information event:  ' Dts.Events.FireInformation(3, "Process Values", "Processing has started", "", 0, fireAgain)  'Example of firing a warning event:  ' Dts.Events.FireWarning(14, "Process Values", "No values received for input", "", 0)  #End Region  #Region "Help: Using Integration Services connection managers in a script"  'Some types of connection managers can be used in this script task. See the topic  '"Working with Connection Managers Programatically" for details.  'Example of using an ADO.Net connection manager:  ' Dim rawConnection As Object = Dts.Connections("Sales DB").AcquireConnection(Dts.Transaction)  ' Dim myADONETConnection As SqlConnection = CType(rawConnection, SqlConnection)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Sales DB").ReleaseConnection(rawConnection)  'Example of using a File connection manager  ' Dim rawConnection As Object = Dts.Connections("Prices.zip").AcquireConnection(Dts.Transaction)  ' Dim filePath As String = CType(rawConnection, String)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Prices.zip").ReleaseConnection(rawConnection)  #End Region  'This method is called when this script task executes in the control flow.  'Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.  'To open Help, press F1.  Public Sub Main()  Dim messages As Collections.ArrayList  Try  messages = CType(Dts.Variables("User::errorsMessages").Value, Collections.ArrayList)  Catch ex As Exception  messages = New Collections.ArrayList()  End Try  messages.Add(Dts.Variables("System::ErrorDescription").Value.ToString())  Dts.Variables("User::errorMessages").Value = messages  Dts.TaskResult = ScriptResults.Success  End Sub  #Region "ScriptResults declaration"  'This enum provides a convenient shorthand within the scope of this class for setting the  'result of the script.  'This code was generated automatically.  Enum ScriptResults  Success = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Success  Failure = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Failure  End Enum  #End Region  End Class |  |
|  |  |
|  |  |
| #Region "Help: Introduction to the script task"  'The Script Task allows you to perform virtually any operation that can be accomplished in  'a .Net application within the context of an Integration Services control flow.  'Expand the other regions which have "Help" prefixes for examples of specific ways to use  'Integration Services features within this script task.  #End Region  #Region "Imports"  Imports System  Imports System.Data  Imports System.Math  Imports Microsoft.SqlServer.Dts.Runtime  #End Region  'ScriptMain is the entry point class of the script. Do not change the name, attributes,  'or parent of this class.  <Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute()> \_  <System.CLSCompliantAttribute(False)> \_  Partial Public Class ScriptMain  Inherits Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase  #Region "Help: Using Integration Services variables and parameters in a script"  'To use a variable in this script, first ensure that the variable has been added to  'either the list contained in the ReadOnlyVariables property or the list contained in  'the ReadWriteVariables property of this script task, according to whether or not your  'code needs to write to the variable. To add the variable, save this script, close this instance of  'Visual Studio, and update the ReadOnlyVariables and  'ReadWriteVariables properties in the Script Transformation Editor window.  'To use a parameter in this script, follow the same steps. Parameters are always read-only.  'Example of reading from a variable:  ' startTime = Dts.Variables("System::StartTime").Value  'Example of writing to a variable:  ' Dts.Variables("User::myStringVariable").Value = "new value"  'Example of reading from a package parameter:  ' batchId = Dts.Variables("$Package::batchId").Value  'Example of reading from a project parameter:  ' batchId = Dts.Variables("$Project::batchId").Value  'Example of reading from a sensitive project parameter:  ' batchId = Dts.Variables("$Project::batchId").GetSensitiveValue()  #End Region  #Region "Help: Firing Integration Services events from a script"  'This script task can fire events for logging purposes.  'Example of firing an error event:  ' Dts.Events.FireError(18, "Process Values", "Bad value", "", 0)  'Example of firing an information event:  ' Dts.Events.FireInformation(3, "Process Values", "Processing has started", "", 0, fireAgain)  'Example of firing a warning event:  ' Dts.Events.FireWarning(14, "Process Values", "No values received for input", "", 0)  #End Region  #Region "Help: Using Integration Services connection managers in a script"  'Some types of connection managers can be used in this script task. See the topic  '"Working with Connection Managers Programatically" for details.  'Example of using an ADO.Net connection manager:  ' Dim rawConnection As Object = Dts.Connections("Sales DB").AcquireConnection(Dts.Transaction)  ' Dim myADONETConnection As SqlConnection = CType(rawConnection, SqlConnection)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Sales DB").ReleaseConnection(rawConnection)  'Example of using a File connection manager  ' Dim rawConnection As Object = Dts.Connections("Prices.zip").AcquireConnection(Dts.Transaction)  ' Dim filePath As String = CType(rawConnection, String)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Prices.zip").ReleaseConnection(rawConnection)  #End Region  'This method is called when this script task executes in the control flow.  'Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.  'To open Help, press F1.  Public Sub Main()  Dim messages As Collections.ArrayList  Try  messages = CType(Dts.Variables("errorTasks").Value, Collections.ArrayList)  Catch ex As Exception  messages = New Collections.ArrayList()  End Try  messages.Add(Dts.Variables("SourceName").Value.ToString() + " failed at " + Dts.Variables("EventHandlerStartTime").Value.ToString())  Dts.Variables("errorTasks").Value = messages  Dts.TaskResult = ScriptResults.Success  End Sub  #Region "ScriptResults declaration"  'This enum provides a convenient shorthand within the scope of this class for setting the  'result of the script.  'This code was generated automatically.  Enum ScriptResults  Success = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Success  Failure = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Failure  End Enum  #End Region  End Class |  |
|  |  |
|  |  |
| #Region "Help: Introduction to the script task"  'The Script Task allows you to perform virtually any operation that can be accomplished in  'a .Net application within the context of an Integration Services control flow.  'Expand the other regions which have "Help" prefixes for examples of specific ways to use  'Integration Services features within this script task.  #End Region  #Region "Imports"  Imports System  Imports System.Data  Imports System.Math  Imports Microsoft.SqlServer.Dts.Runtime  #End Region  'ScriptMain is the entry point class of the script. Do not change the name, attributes,  'or parent of this class.  <Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute()> \_  <System.CLSCompliantAttribute(False)> \_  Partial Public Class ScriptMain  Inherits Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase  #Region "Help: Using Integration Services variables and parameters in a script"  'To use a variable in this script, first ensure that the variable has been added to  'either the list contained in the ReadOnlyVariables property or the list contained in  'the ReadWriteVariables property of this script task, according to whether or not your  'code needs to write to the variable. To add the variable, save this script, close this instance of  'Visual Studio, and update the ReadOnlyVariables and  'ReadWriteVariables properties in the Script Transformation Editor window.  'To use a parameter in this script, follow the same steps. Parameters are always read-only.  'Example of reading from a variable:  ' startTime = Dts.Variables("System::StartTime").Value  'Example of writing to a variable:  ' Dts.Variables("User::myStringVariable").Value = "new value"  'Example of reading from a package parameter:  ' batchId = Dts.Variables("$Package::batchId").Value  'Example of reading from a project parameter:  ' batchId = Dts.Variables("$Project::batchId").Value  'Example of reading from a sensitive project parameter:  ' batchId = Dts.Variables("$Project::batchId").GetSensitiveValue()  #End Region  #Region "Help: Firing Integration Services events from a script"  'This script task can fire events for logging purposes.  'Example of firing an error event:  ' Dts.Events.FireError(18, "Process Values", "Bad value", "", 0)  'Example of firing an information event:  ' Dts.Events.FireInformation(3, "Process Values", "Processing has started", "", 0, fireAgain)  'Example of firing a warning event:  ' Dts.Events.FireWarning(14, "Process Values", "No values received for input", "", 0)  #End Region  #Region "Help: Using Integration Services connection managers in a script"  'Some types of connection managers can be used in this script task. See the topic  '"Working with Connection Managers Programatically" for details.  'Example of using an ADO.Net connection manager:  ' Dim rawConnection As Object = Dts.Connections("Sales DB").AcquireConnection(Dts.Transaction)  ' Dim myADONETConnection As SqlConnection = CType(rawConnection, SqlConnection)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Sales DB").ReleaseConnection(rawConnection)  'Example of using a File connection manager  ' Dim rawConnection As Object = Dts.Connections("Prices.zip").AcquireConnection(Dts.Transaction)  ' Dim filePath As String = CType(rawConnection, String)  ' <Use the connection in some code here, then release the connection>  ' Dts.Connections("Prices.zip").ReleaseConnection(rawConnection)  #End Region  'This method is called when this script task executes in the control flow.  'Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.  'To open Help, press F1.  Public Sub Main()  Dim errorDesc As String  Dim messages As Collections.ArrayList  Try  messages = CType(Dts.Variables("errorTasks").Value, Collections.ArrayList)  Catch ex As Exception  'If there is an exception ? the object was never initialized, so there were no errors  Return  End Try  For Each errorDesc In messages  Dts.Variables("errorString").Value = Dts.Variables("errorString").Value.ToString + errorDesc + vbCrLf  Next  Try  messages = CType(Dts.Variables("errorMessages").Value, Collections.ArrayList)  Catch ex As Exception  'If there is an exception ? the object was never initialized, so there were no errors  Return  End Try  messages.Sort()  For Each errorDesc In messages  Dts.Variables("errorString").Value = Dts.Variables("errorString").Value.ToString + errorDesc + vbCrLf  Next  Dts.TaskResult = ScriptResults.Success  End Sub  #Region "ScriptResults declaration"  'This enum provides a convenient shorthand within the scope of this class for setting the  'result of the script.  'This code was generated automatically.  Enum ScriptResults  Success = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Success  Failure = Microsoft.SqlServer.Dts.Runtime.DTSExecResult.Failure  End Enum  #End Region  End Class |  |