**Similarities between the Script Task and the Script Component**

The Script task and the Script component share the following common features.

|  |  |
| --- | --- |
| **Feature** | **Description** |
| Two design-time modes | In both the task and the component, you begin by specifying properties in the editor, and then switch to the development environment to write code. |
| Microsoft Visual Studio Tools for Applications (VSTA) | Both the task and the component use the same VSTA IDE, and support code written in either Microsoft Visual Basic 2008 or Microsoft Visual C# 2008. |
| Precompiled scripts | In SQL Server 2008 Integration Services (SSIS), all scripts are precompiled. In earlier versions, you could specify whether scripts were precompiled.  The script is precompiled into binary code, permitting faster execution, but at the cost of increased package size. |

**ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/collapse.gifDifferences between the Script Task and the Script Component**

The Script task and the Script component have the following noteworthy differences.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Script Task** | **Script Component** |
| Control flow / Data flow | The Script task is configured on the Control Flow tab of the designer and runs outside the data flow of the package. | The Script component is configured on the Data Flow page of the designer and represents a source, transformation, or destination in the Data Flow task. |
| Purpose | A Script task can accomplish almost any general-purpose task. | You must specify whether you want to create a source, transformation, or destination with the Script component. |
| Execution | A Script task runs custom code at some point in the package workflow. Unless you put it in a loop container or an event handler, it only runs once. | A Script component also runs once, but typically it runs its main processing routine once for each row of data in the data flow. |
| Editor | The **Script Task Editor** has three pages: **General**, **Script**, and **Expressions**. Only the **ReadOnlyVariables** and **ReadWriteVariables**, and **ScriptLanguage** properties directly affect the code that you can write. | The **Script Transformation Editor** has up to four pages: **Input Columns**, **Inputs and Outputs**, **Script**, and **Connection Managers**. The metadata and properties that you configure on each of these pages determines the members of the base classes that are autogenerated for your use in coding. |
| Interaction with the package | In the code written for a Script task, you use the **Dts** property to access other features of the package. The **Dts** property is a member of the **ScriptMain** class. | In Script component code, you use typed accessor properties to access certain package features such as variables and connection managers.  The **PreExecute** method can access only read-only variables. The **PostExecute** method can access both read-only and read/write variables. |
| Using variables | The Script task uses the Variables property of the **Dts** object to access variables that are available through the task’s ReadOnlyVariables and ReadWriteVariables properties. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim myVar as String  myVar = Dts.Variables(“MyStringVariable”).Value.ToString | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | string myVar;  myVar = Dts.Variables(MyStringVariable).Value.ToString; | | | | The Script component uses typed accessor properties of the autogenerated based class, created from the component’s ReadOnlyVariables and ReadWriteVariables properties. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim myVar as String  myVar = Me.Variables.MyStringVariable | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | string myVar;  myVar = this.Variables.MyStringVariable; | | | |
| Using connections | The Script task uses the Connections property of the **Dts** object to access connection managers defined in the package. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim myFlatFileConnection As String  myFlatFileConnection = \_  DirectCast(Dts.Connections("Test Flat File Connection").AcquireConnection(Dts.Transaction), \_  String) | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | string myFlatFileConnection;  myFlatFileConnection = DirectCast(Dts.Connections("Test Flat File Connection").AcquireConnection(Dts.Transaction), String); | | | | The Script component uses typed accessor properties of the autogenerated base class, created from the list of connection managers entered by the user on the Connection Managers page of the editor. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim connMgr As IDTSConnectionManager100  connMgr = Me.Connections.MyADONETConnection | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | IDTSConnectionManager100 connMgr;  connMgr = this.Connections.MyADONETConnection; | | | |
| Raising events | The Script task uses the Events property of the **Dts** object to raise events. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dts.Events.FireError(0, "Event Snippet", \_  ex.Message & ControlChars.CrLf & ex.StackTrace, \_  "", 0) | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | Dts.Events.FireError(0, "Event Snippet", ex.Message+ControlChars.CrLf+ex.StackTrace, "", 0); | | | | The Script component raises errors, warnings, and informational messages by using the methods of the IDTSComponentMetaData100 interface returned by the ComponentMetaData property. For example:   |  |  |  | | --- | --- | --- | | **Visual Basic** | | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim myMetadata as IDTSComponentMetaData100  myMetaData = Me.ComponentMetaData  myMetaData.FireError(...) | | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | | IDTSComponentMetaData100 myMetadata;  myMetaData = this.ComponentMetaData;  myMetaData.FireError(...); | | | |
| Logging | The Script task uses the Log method of the **Dts** object to log information to enabled log providers. For example:   |  |  | | --- | --- | | **Visual Basic** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | Dim bt(0) As Byte  Dts.Log("Test Log Event", \_  0, \_  bt) | | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | byte[] bt = new byte[0];  Dts.Log("Test Log Event", 0, bt); | | | The Script component uses the Log method of the autogenerated base class to log information to enabled log providers. For example:  [Visual Basic]  Dim bt(0) As Byte  Me.Log("Test Log Event", \_  0, \_  bt)   |  |  | | --- | --- | | **C#** | ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/copycode.gif | | byte[] bt = new byte[0];  this.Log("Test Log Event", 0, bt); | | |
| Returning results | The Script task uses both the TaskResult property and the optional ExecutionValue property of the **Dts** object to notify the runtime of its results. | The Script component runs as a part of the Data Flow task and does not report results using either of these properties. |
| Debugging | The Script task supports breakpoints and stepping through code while debugging in the design environment.   |  | | --- | | **ms-help://MS.SQLCC.v10/MS.SQLSVR.v10.en/s10is_2devguide/local/note.gifNote:** | | When you debug a package that contains multiple Script tasks, the debugger hits breakpoints in only one Script task and will ignore breakpoints in the other Script tasks. If a Script task is part of a Foreach Loop or For Loop container, the debugger ignores breakpoints in the Script task after the first iteration of the loop. | | The Script component does not support debugging |