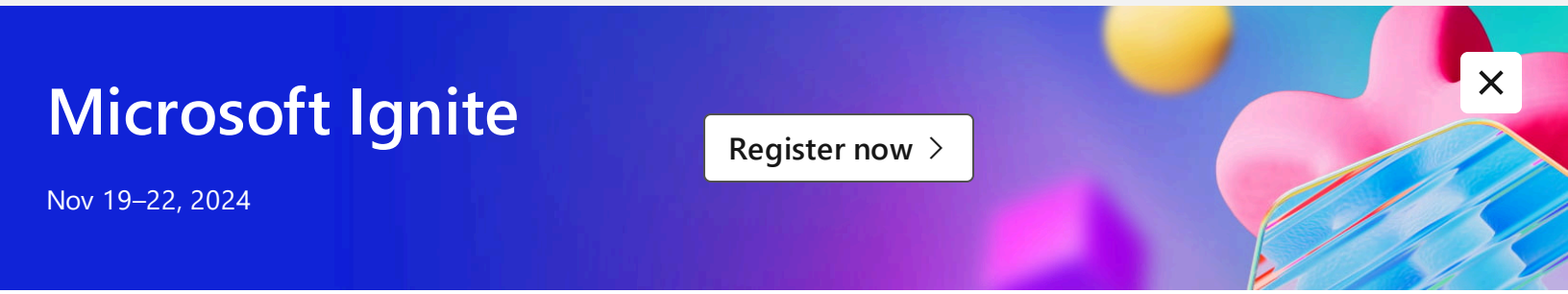

Type.GetTypeFromCLSID Method

Reference

In this article

Definition
Overloads
GetTypeFromCLSID(Guid)
GetTypeFromCLSID(Guid, Boolean)

Show 2 more

Definition

Namespace: System

Assembly: System.Runtime.dll

Gets the type associated with the specified class identifier (CLSID).

Overloads

Expand table

GetTypeFromCLSID(Guid)	Gets the type associated with the specified class identifier (CLSID).
GetTypeFromCLSID(Guid, Boolean)	Gets the type associated with the specified class identifier (CLSID), specifying whether to throw an exception if an error occurs while loading the type.
GetTypeFromCLSID(Guid, String)	Gets the type associated with the specified class identifier (CLSID) from the specified server.
GetTypeFromCLSID(Guid, String, Boolean)	Gets the type associated with the specified class identifier (CLSID) from the specified server, specifying whether to throw an exception if an error occurs while loading the type.

GetTypeFromCLSID(Guid)

Source: Type.cs

Gets the type associated with the specified class identifier (CLSID).

```
public:
 static Type ^ GetTypeFromCLSID(Guid clsid);
```

```
[System.Runtime.Versioning.SupportedOSPlatform("windows")
public static Type? GetTypeFromCLSID (Guid clsid);
```

```
[<System.Runtime.Versioning.SupportedOSPlatform("windows'
static member GetTypeFromCLSID : Guid -> Type
```

```
Public Shared Function GetTypeFromCLSID (clsid As
Guid) As Type
```

Parameters

`clsid` Guid

The CLSID of the type to get.

Returns

Type

`System.__ComObject` regardless of whether the CLSID is valid.

Attributes SupportedOSPlatformAttribute

Examples

The following example uses the CLSID of the Microsoft Word Application object to retrieve a COM type that represents the Microsoft Word application. It then instantiates the type by calling the Activator.CreateInstance method, and closes it by calling the Application.Quit method.

```
using System;
using System.Reflection;
using System.Runtime.InteropServices;

public class Example
{
   private const string WORD_CLSID = "{000209FF-0000-0000

   public static void Main()
   {
      // Start an instance of the Word application.
      var word = Type.GetTypeFromCLSID(Guid.Parse(WORD_CL
      Console.WriteLine("Instantiated Type object from CL
                        WORD_CLSID);
      Object wordObj = Activator.CreateInstance(word);
      Console.WriteLine("Instantiated {0}",
                        wordObj.GetType().FullName);

      // Close Word.
      word.InvokeMember("Quit", BindingFlags.InvokeMethod
                        wordObj, new object[] { 0, 0, fal
   }
}
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
open System
open System.Reflection

let [<Literal>] WORD_CLSID = "{000209FF-0000-0000-C000-00

// Start an instance of the Word application.
let word = Type.GetTypeFromCLSID(Guid.Parse WORD_CLSID)
```

```
printfn $"Instantiated Type object from CLSID {WORD_CLSID
let wordObj = Activator.CreateInstance word
printfn $"Instantiated {wordObj.GetType().FullName}"

// Close Word.
word.InvokeMember("Quit", BindingFlags.InvokeMethod, null
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
Imports System.Reflection
Imports System.Runtime.InteropServices

Module Example
   Private Const WORD_CLSID As String = "{000209FF-0000-0

   Public Sub Main()
      ' Start an instance of the Word application.
      Dim word As Type = Type.GetTypeFromCLSID(Guid.Parse
      Console.WriteLine("Instantiated Type object from CL
                        WORD_CLSID)
      Dim wordObj As Object = Activator.CreateInstance(wo
      Console.WriteLine("Instantiated {0}",
                        wordObj.GetType().FullName)

      ' Close Word.
      word.InvokeMember("Quit", BindingFlags.InvokeMethod
                        wordObj, New Object() { 0, 0, Fal
   End Sub
End Module
' The example displays the following output:
'    Instantiated Type object from CLSID {000209FF-0000-0
'    Instantiated Microsoft.Office.Interop.Word.Applicati
```

Remarks

The GetTypeFromCLSID method supports late-bound access to unmanaged COM objects from .NET Framework apps when you know the COM object's class identifier (CLSID). The class identifier for COM classes is defined in the HKEY_CLASSES_ROOT\CLSID key

of the registry. You can retrieve the value of the IsCOMObject property to determine whether the type returned by this method is a COM object.

> **Tip**
>
> You can call the **GetTypeFromProgID** method for late-bound access to COM objects whose programmatic identifier (ProgID) you know.

Instantiating an unmanaged COM object from its CLSID is a two-step process:

1. Get a Type object that represents the `__ComObject` that corresponds to the CLSID by calling the GetTypeFromCLSID method.
2. Call the Activator.CreateInstance(Type) method to instantiate the COM object.

See the example for an illustration.

The GetTypeFromCLSID(Guid) overload ignores any exception that may occur when instantiating a Type object based on the `clsid` argument. Note that no exception is thrown if `clsid` is not found in the registry.

Notes to Callers

This method is intended for use when working with COM objects, not with .NET Framework objects. All managed objects, including those that are visible to COM (that is, their ComVisibleAttribute attribute is `true`) have a GUID that is returned by the GUID property. Although the method returns a Type object that corresponds to the GUID for .NET Framework objects, you can't use

that Type object to create a type instance by calling the CreateInstance(Type) method, as the following example shows.

```
using System;
using System.Runtime.InteropServices;

[assembly:ComVisible(true)]

// Define two classes, and assign one an explicit GUID.
[GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")]
public class ExplicitGuid
{ }

public class NoExplicitGuid
{ }

public class Example
{
   public static void Main()
   {
      Type explicitType = typeof(ExplicitGuid);
      Guid explicitGuid = explicitType.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type explicitCOM = Type.GetTypeFromCLSID(explicitGu
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid);

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(explicitCO
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }

      Type notExplicit = typeof(NoExplicitGuid);
      Guid notExplicitGuid = notExplicit.GUID;
```

```
      // Get type of ExplicitGuid from its GUID.
      Type notExplicitCOM = Type.GetTypeFromCLSID(notExpl
      Console.WriteLine("Created {0} type from CLSID {1}'
                        notExplicitCOM.Name, notExplicitC

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(notExplici
         Console.WriteLine("Instantiated a {0} object", c
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }
   }
}
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
open System
open System.Runtime.InteropServices

[<assembly: ComVisible true>]
do ()

// Define two classes, and assign one an explicit GUID.
```

```
[<Guid "d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4">]
type ExplicitGuid() = class end

type NoExplicitGuid() = class end

let explicitType = typeof<ExplicitGuid>
let explicitGuid = explicitType.GUID

// Get type of ExplicitGuid from its GUID.
let explicitCOM = Type.GetTypeFromCLSID explicitGuid
printfn $"Created {explicitCOM.Name} type from CLSID {exp

// Compare the two type objects.
printfn $"{explicitType.Name} and {explicitCOM.Name} equa

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance explicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"

let notExplicit = typeof<NoExplicitGuid>
let notExplicitGuid = notExplicit.GUID

// Get type of ExplicitGuid from its GUID.
let notExplicitCOM = Type.GetTypeFromCLSID(notExplicitGui
printfn $"Created {notExplicitCOM.Name} type from CLSID {

// Compare the two type objects.
printfn $"{notExplicit.Name} and {notExplicitCOM.Name} eq

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance notExplicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
```

```
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
Imports System.Runtime.InteropServices

<Assembly:ComVisible(True)>

' Define two classes, and assign one an explicit GUID.
<GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")>
Public Class ExplicitGuid
End Class

Public Class NoExplicitGuid
End Class

Module Example
   Public Sub Main()
      Dim explicitType As Type = GetType(ExplicitGuid)
      Dim explicitGuid As Guid = explicitType.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim explicitCOM As Type = Type.GetTypeFromCLSID(exp
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid)

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(exp
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try

      Dim notExplicit As Type = GetType(NoExplicitGuid)
      Dim notExplicitGuid As Guid = notExplicit.GUID
```

```
      ' Get type of ExplicitGuid from its GUID.
      Dim notExplicitCOM As Type = Type.GetTypeFromCLSID(
      Console.WriteLine("Created {0} type from CLSID {1}'
                        notExplicitCOM.Name, notExplicitC

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(not
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try
   End Sub
End Module
' The example displays the following output:
'       Created __ComObject type from CLSID d055cba3-1f83
'       ExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
'
'       Created __ComObject type from CLSID 74f03346-a718
'       NoExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {74F03346-A718-3516-AC78-F351C7459FFB} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
```

Instead, the GetTypeFromCLSID(Guid, String, Boolean) should only be used to retrieve the GUID of an unmanaged COM object, and the resulting Type object that is passed to the CreateInstance(Type) method must represent an unmanaged COM object.

Applies to

▼

GetTypeFromCLSID(Guid, Boolean)

Source: Type.cs

Gets the type associated with the specified class identifier (CLSID), specifying whether to throw an exception if an error occurs while loading the type.

```
public:
 static Type ^ GetTypeFromCLSID(Guid clsid, bool
throwOnError);
```

```
[System.Runtime.Versioning.SupportedOSPlatform("windows")
public static Type? GetTypeFromCLSID (Guid clsid, bool
throwOnError);
```

```
[<System.Runtime.Versioning.SupportedOSPlatform("windows'
static member GetTypeFromCLSID : Guid * bool -> Type
```

```
Public Shared Function GetTypeFromCLSID (clsid As
Guid, throwOnError As Boolean) As Type
```

Parameters

`clsid` Guid

The CLSID of the type to get.

`throwOnError` Boolean

`true` to throw any exception that occurs.

-or-

`false` to ignore any exception that occurs.

Returns

Type

`System.__ComObject` regardless of whether the CLSID is valid.

Attributes SupportedOSPlatformAttribute

Examples

The following example uses the CLSID of the Microsoft Word Application object to retrieve a COM type that represents the Microsoft Word application. It then instantiates the type by calling the Activator.CreateInstance method, and closes it by calling the Application.Quit method. An exception is thrown if an error occurs while loading the type.

```
using System;
using System.Reflection;
using System.Runtime.InteropServices;

public class Example
{
   private const string WORD_CLSID = "{000209FF-0000-0000

   public static void Main()
   {
      try {
         // Start an instance of the Word application.
         var word = Type.GetTypeFromCLSID(Guid.Parse(WORD
         Console.WriteLine("Instantiated Type object from
```

```
                              WORD_CLSID);
         Object wordObj = Activator.CreateInstance(word);
         Console.WriteLine("Instantiated {0}",
                              wordObj.GetType().FullName, WO

         // Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                              wordObj, new object[] { 0, 0,
      }
      catch (Exception) {
         Console.WriteLine("Unable to instantiate an obje
      }
   }
}
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
open System
open System.Reflection

let [<Literal>] WORD_CLSID = "{000209FF-0000-0000-C000-00

try
    // Start an instance of the Word application.
    let word = Type.GetTypeFromCLSID(Guid.Parse WORD_CLSI
    printfn $"Instantiated Type object from CLSID {WORD_C
    let wordObj = Activator.CreateInstance word
    printfn $"Instantiated {wordObj.GetType().FullName} f

    // Close Word.
    word.InvokeMember("Quit", BindingFlags.InvokeMethod,
with _ ->
    printfn $"Unable to instantiate an object for {WORD_C
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
Imports System.Reflection
Imports System.Runtime.InteropServices
```

```
Module Example
   Private Const WORD_CLSID As String = "{000209FF-0000-0
   Public Sub Main()
      ' Start an instance of the Word application.
      Try
         Dim word As Type = Type.GetTypeFromCLSID(Guid.Pa
         Console.WriteLine("Instantiated Type object from
                           WORD_CLSID)
         Dim wordObj As Object = Activator.CreateInstance
         Console.WriteLine("Instantiated {0}",
                           wordObj.GetType().FullName)

         ' Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                           wordObj, New Object() { 0, 0,
      ' The method can throw any of a number of unexpecte
      Catch e As Exception
         Console.WriteLine("Unable to instantiate an obje
      End Try
   End Sub
End Module
' The example displays the following output:
'    Instantiated Type object from CLSID {000209FF-0000-0
'    Instantiated Microsoft.Office.Interop.Word.Applicati
```

Remarks

The GetTypeFromCLSID method supports late-bound access to unmanaged COM objects from .NET Framework apps when you know the COM object's class identifier (CLSID). The class identifier for COM classes is defined in the HKEY_CLASSES_ROOT\CLSID key of the registry. You can retrieve the value of the IsCOMObject property to determine whether the type returned by this method is a COM object.

Tip

> You can call the **GetTypeFromProgID** method for late-bound access to COM objects whose programmatic identifier (ProgID) you know.

Instantiating an unmanaged COM object from its CLSID is a two-step process:

1. Get a Type object that represents the `__ComObject` that corresponds to the CLSID by calling the GetTypeFromCLSID method.
2. Call the Activator.CreateInstance(Type) method to instantiate the COM object.

See the example for an illustration.

Exceptions such as OutOfMemoryException will be thrown when specifying `true` for `throwOnError`, but it will not fail for unregistered CLSIDs.

Notes to Callers

This method is intended for use when working with COM objects, not with .NET Framework objects. All managed objects, including those that are visible to COM (that is, their ComVisibleAttribute attribute is `true`) have a GUID that is returned by the GUID property. Although the method returns a Type object that corresponds to the GUID for .NET Framework objects, you can't use that Type object to create a type instance by calling the CreateInstance(Type) method, as the following example shows.

```
using System;
using System.Runtime.InteropServices;

[assembly:ComVisible(true)]
```

```
// Define two classes, and assign one an explicit GUID.
[GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")]
public class ExplicitGuid
{ }

public class NoExplicitGuid
{ }

public class Example
{
   public static void Main()
   {
      Type explicitType = typeof(ExplicitGuid);
      Guid explicitGuid = explicitType.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type explicitCOM = Type.GetTypeFromCLSID(explicitGu
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid);

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(explicitCO
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }

      Type notExplicit = typeof(NoExplicitGuid);
      Guid notExplicitGuid = notExplicit.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type notExplicitCOM = Type.GetTypeFromCLSID(notExpl
      Console.WriteLine("Created {0} type from CLSID {1}'
                        notExplicitCOM.Name, notExplicitG

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      // Instantiate an ExplicitGuid object.
```

```
      try {
         Object obj = Activator.CreateInstance(notExplici
         Console.WriteLine("Instantiated a {0} object", c
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }
   }
}
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
open System
open System.Runtime.InteropServices

[<assembly: ComVisible true>]
do ()

// Define two classes, and assign one an explicit GUID.
[<Guid "d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4">]
type ExplicitGuid() = class end

type NoExplicitGuid() = class end

let explicitType = typeof<ExplicitGuid>
let explicitGuid = explicitType.GUID

// Get type of ExplicitGuid from its GUID.
let explicitCOM = Type.GetTypeFromCLSID explicitGuid
printfn $"Created {explicitCOM.Name} type from CLSID {exp
```

```
// Compare the two type objects.
printfn $"{explicitType.Name} and {explicitCOM.Name} equa

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance explicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"

let notExplicit = typeof<NoExplicitGuid>
let notExplicitGuid = notExplicit.GUID

// Get type of ExplicitGuid from its GUID.
let notExplicitCOM = Type.GetTypeFromCLSID(notExplicitGui
printfn $"Created {notExplicitCOM.Name} type from CLSID {

// Compare the two type objects.
printfn $"{notExplicit.Name} and {notExplicitCOM.Name} eq

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance notExplicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
Imports System.Runtime.InteropServices

<Assembly:ComVisible(True)>

' Define two classes, and assign one an explicit GUID.
<GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")>
Public Class ExplicitGuid
End Class

Public Class NoExplicitGuid
End Class

Module Example
   Public Sub Main()
      Dim explicitType As Type = GetType(ExplicitGuid)
      Dim explicitGuid As Guid = explicitType.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim explicitCOM As Type = Type.GetTypeFromCLSID(exp
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid)

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(exp
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try

      Dim notExplicit As Type = GetType(NoExplicitGuid)
      Dim notExplicitGuid As Guid = notExplicit.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim notExplicitCOM As Type = Type.GetTypeFromCLSID(
      Console.WriteLine("Created {0} type from CLSID {1}'
                        notExplicitCOM.Name, notExplicitC

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM
```

```
        ' Instantiate an ExplicitGuid object.
        Try
           Dim obj As Object = Activator.CreateInstance(not
           Console.WriteLine("Instantiated a {0} object", c
        Catch e As COMException
           Console.WriteLine("COM Exception:{1}{0}{1}", e.M
        End Try
    End Sub
End Module
' The example displays the following output:
'       Created __ComObject type from CLSID d055cba3-1f83
'       ExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
'
'       Created __ComObject type from CLSID 74f03346-a718
'       NoExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {74F03346-A718-3516-AC78-F351C7459FFB} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
```

Instead, the GetTypeFromCLSID(Guid, String, Boolean) should only be used to retrieve the GUID of an unmanaged COM object, and the resulting Type object that is passed to the CreateInstance(Type) method must represent an unmanaged COM object.

Applies to

▼

GetTypeFromCLSID(Guid, String)

Source: Type.cs

Gets the type associated with the specified class identifier (CLSID) from the specified server.

```
public:
 static Type ^ GetTypeFromCLSID(Guid clsid,
System::String ^ server);
```

```
[System.Runtime.Versioning.SupportedOSPlatform("windows")
public static Type? GetTypeFromCLSID (Guid clsid,
string? server);
```

```
[<System.Runtime.Versioning.SupportedOSPlatform("windows'
static member GetTypeFromCLSID : Guid * string -> Type
```

```
Public Shared Function GetTypeFromCLSID (clsid As
Guid, server As String) As Type
```

Parameters

`clsid` Guid

The CLSID of the type to get.

`server` String

The server from which to load the type. If the server name is `null`, this method automatically reverts to the local machine.

Returns

Type

`System.__ComObject` regardless of whether the CLSID is valid.

Attributes SupportedOSPlatformAttribute

Examples

The following example uses the CLSID of the Microsoft Word Application object to retrieve a COM type that represents the Microsoft Word application from a server named computer17.central.contoso.com. It then instantiates the type by calling the Activator.CreateInstance method, and closes it by calling the Application.Quit method.

```
using System;
using System.Reflection;
using System.Runtime.InteropServices;

public class Example
{
   private const string WORD_CLSID = "{000209FF-0000-0000

   public static void Main()
   {
      // Start an instance of the Word application.
      var word = Type.GetTypeFromCLSID(Guid.Parse(WORD_CL
      Console.WriteLine("Instantiated Type object from CL
                        WORD_CLSID);
      try {
         Object wordObj = Activator.CreateInstance(word);
         Console.WriteLine("Instantiated {0}",
                           wordObj.GetType().FullName, WO

         // Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                           wordObj, new object[] { 0, 0,
      }
      catch (COMException) {
         Console.WriteLine("Unable to instantiate object.
      }
   }
}
```

```
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
open System
open System.Reflection
open System.Runtime.InteropServices

let [<Literal>] WORD_CLSID = "{000209FF-0000-0000-C000-00

// Start an instance of the Word application.
let word = Type.GetTypeFromCLSID(Guid.Parse WORD_CLSID, '
printfn $"Instantiated Type object from CLSID {WORD_CLSID
try
    let wordObj = Activator.CreateInstance word
    printfn $"Instantiated {wordObj.GetType().FullName} f

    // Close Word.
    word.InvokeMember("Quit", BindingFlags.InvokeMethod,
with :? COMException ->
    printfn "Unable to instantiate object."
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
Imports System.Reflection
Imports System.Runtime.InteropServices

Module Example
   Private Const WORD_CLSID As String = "{000209FF-0000-0

   Public Sub Main()
      ' Start an instance of the Word application.
      Dim word As Type = Type.GetTypeFromCLSID(Guid.Parse
      Console.WriteLine("Instantiated Type object from CL
                        WORD_CLSID)
      Try
         Dim wordObj As Object = Activator.CreateInstance
         Console.WriteLine("Instantiated {0}",
                           wordObj.GetType().FullName)
```

```
         ' Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                           wordObj, New Object() { 0, 0,
      Catch e As COMException
         Console.WriteLine("Unable to instantiate object.
      End Try
   End Sub
End Module
' The example displays the following output:
'    Instantiated Type object from CLSID {000209FF-0000-0
'    Instantiated Microsoft.Office.Interop.Word.Applicati
```

Remarks

The GetTypeFromCLSID method supports late-bound access to unmanaged COM objects from .NET Framework apps when you know the COM object's class identifier (CLSID). The class identifier for COM classes is defined in the HKEY_CLASSES_ROOT\CLSID key of the registry. You can retrieve the value of the IsCOMObject property to determine whether the type returned by this method is a COM object.

> **Tip**
>
> You can call the **GetTypeFromProgID** method for late-bound access to COM objects whose programmatic identifier (ProgID) you know.

Instantiating an unmanaged COM object from its CLSID is a two-step process:

1. Get a Type object that represents the `__ComObject` that corresponds to the CLSID by calling the GetTypeFromCLSID method.

2. Call the Activator.CreateInstance(Type) method to instantiate the COM object.

Notes to Callers

This method is intended for use when working with COM objects, not with .NET Framework objects. All managed objects, including those that are visible to COM (that is, their ComVisibleAttribute attribute is `true`) have a GUID that is returned by the GUID property. Although the method returns a Type object that corresponds to the GUID for .NET Framework objects, you can't use that Type object to create a type instance by calling the CreateInstance(Type) method, as the following example shows.

```
using System;
using System.Runtime.InteropServices;

[assembly:ComVisible(true)]

// Define two classes, and assign one an explicit GUID.
[GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")]
public class ExplicitGuid
{ }

public class NoExplicitGuid
{ }

public class Example
{
   public static void Main()
   {
      Type explicitType = typeof(ExplicitGuid);
      Guid explicitGuid = explicitType.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type explicitCOM = Type.GetTypeFromCLSID(explicitGu
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid);

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      // Instantiate an ExplicitGuid object.
```

```
      try {
         Object obj = Activator.CreateInstance(explicitCO
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }

      Type notExplicit = typeof(NoExplicitGuid);
      Guid notExplicitGuid = notExplicit.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type notExplicitCOM = Type.GetTypeFromCLSID(notExpl
      Console.WriteLine("Created {0} type from CLSID {1}'
                        notExplicitCOM.Name, notExplicitG

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(notExplici
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }
   }
}
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
open System
open System.Runtime.InteropServices

[<assembly: ComVisible true>]
do ()

// Define two classes, and assign one an explicit GUID.
[<Guid "d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4">]
type ExplicitGuid() = class end

type NoExplicitGuid() = class end

let explicitType = typeof<ExplicitGuid>
let explicitGuid = explicitType.GUID

// Get type of ExplicitGuid from its GUID.
let explicitCOM = Type.GetTypeFromCLSID explicitGuid
printfn $"Created {explicitCOM.Name} type from CLSID {exp

// Compare the two type objects.
printfn $"{explicitType.Name} and {explicitCOM.Name} equa

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance explicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"

let notExplicit = typeof<NoExplicitGuid>
let notExplicitGuid = notExplicit.GUID

// Get type of ExplicitGuid from its GUID.
let notExplicitCOM = Type.GetTypeFromCLSID(notExplicitGui
printfn $"Created {notExplicitCOM.Name} type from CLSID {

// Compare the two type objects.
printfn $"{notExplicit.Name} and {notExplicitCOM.Name} eq

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance notExplicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"
```

```
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
Imports System.Runtime.InteropServices

<Assembly:ComVisible(True)>

' Define two classes, and assign one an explicit GUID.
<GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")>
Public Class ExplicitGuid
End Class

Public Class NoExplicitGuid
End Class

Module Example
   Public Sub Main()
      Dim explicitType As Type = GetType(ExplicitGuid)
      Dim explicitGuid As Guid = explicitType.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim explicitCOM As Type = Type.GetTypeFromCLSID(exp
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid)

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))
```

```
      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(exp
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try

      Dim notExplicit As Type = GetType(NoExplicitGuid)
      Dim notExplicitGuid As Guid = notExplicit.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim notExplicitCOM As Type = Type.GetTypeFromCLSID(
      Console.WriteLine("Created {0} type from CLSID {1}"
                        notExplicitCOM.Name, notExplicitC

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(not
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try
   End Sub
End Module
' The example displays the following output:
'       Created __ComObject type from CLSID d055cba3-1f83
'       ExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
'
'       Created __ComObject type from CLSID 74f03346-a718
'       NoExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {74F03346-A718-3516-AC78-F351C7459FFB} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
```

Instead, the GetTypeFromCLSID(Guid, String, Boolean) should only be used to retrieve the GUID of an unmanaged COM object, and the resulting Type object that is passed to the CreateInstance(Type) method must represent an unmanaged COM object.

Applies to

▼

GetTypeFromCLSID(Guid, String, Boolean)

Source: Type.cs

Gets the type associated with the specified class identifier (CLSID) from the specified server, specifying whether to throw an exception if an error occurs while loading the type.

```
public:
 static Type ^ GetTypeFromCLSID(Guid clsid,
System::String ^ server, bool throwOnError);
```

```
[System.Runtime.Versioning.SupportedOSPlatform("windows")
public static Type? GetTypeFromCLSID (Guid clsid,
string? server, bool throwOnError);
```

```
[<System.Runtime.Versioning.SupportedOSPlatform("windows'
static member GetTypeFromCLSID : Guid * string * bool
-> Type
```

```
Public Shared Function GetTypeFromCLSID (clsid As
Guid, server As String, throwOnError As Boolean) As
Type
```

Parameters

`clsid` Guid

The CLSID of the type to get.

`server` String

The server from which to load the type. If the server name is `null`, this method automatically reverts to the local machine.

`throwOnError` Boolean

`true` to throw any exception that occurs.

-or-

`false` to ignore any exception that occurs.

Returns

Type

`System.__ComObject` regardless of whether the CLSID is valid.

Attributes SupportedOSPlatformAttribute

Examples

The following example uses the CLSID of the Microsoft Word Application object to retrieve a COM type that represents the Microsoft Word application from a server named computer17.central.contoso.com. It then instantiates the type by

calling the Activator.CreateInstance method, and closes it by calling the Application.Quit method. An exception is thrown if an error occurs while loading the type.

```
using System;
using System.Reflection;
using System.Runtime.InteropServices;

public class Example
{
   private const string WORD_CLSID = "{000209FF-0000-0000

   public static void Main()
   {
      try {
         // Start an instance of the Word application.
         var word = Type.GetTypeFromCLSID(Guid.Parse(WORD
                                          "computer17.cer
                                          true);
         Console.WriteLine("Instantiated Type object from
                           WORD_CLSID);
         Object wordObj = Activator.CreateInstance(word);
         Console.WriteLine("Instantiated {0}",
                           wordObj.GetType().FullName, WO

         // Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                           wordObj, new object[] { 0, 0,
      }
      // The method can throw any of a variety of excepti
      catch (Exception e) {
         Console.WriteLine("{0}: Unable to instantiate ar
                           e.GetType().Name, WORD_CLSID);
      }
   }
}
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
open System
open System.Reflection

let [<Literal>] WORD_CLSID = "{000209FF-0000-0000-C000-00

try
    // Start an instance of the Word application.
    let word = Type.GetTypeFromCLSID(Guid.Parse WORD_CLSI
    printfn $"Instantiated Type object from CLSID {WORD_C
    let wordObj = Activator.CreateInstance word
    printfn $"Instantiated {wordObj.GetType().FullName} f

    // Close Word.
    word.InvokeMember("Quit", BindingFlags.InvokeMethod,
// The method can throw any of a variety of exceptions.
with e ->
    printfn $"{e.GetType().Name}: Unable to instantiate a
// The example displays the following output:
//    Instantiated Type object from CLSID {000209FF-0000-
//    Instantiated Microsoft.Office.Interop.Word.Applicat
```

```
Imports System.Reflection
Imports System.Runtime.InteropServices

Module Example
   Private Const WORD_CLSID As String = "{000209FF-0000-0

   Public Sub Main()
      Try
         ' Start an instance of the Word application.
         Dim word As Type = Type.GetTypeFromCLSID(Guid.Pa
                                                  "comput
                                                  True)
         Console.WriteLine("Instantiated Type object from
                           WORD_CLSID)

         Dim wordObj As Object = Activator.CreateInstance
         Console.WriteLine("Instantiated {0}",
                           wordObj.GetType().FullName)

         ' Close Word.
         word.InvokeMember("Quit", BindingFlags.InvokeMet
                           wordObj, New Object() { 0, 0,
      ' The method can throw any of a variety of exceptio
```

```
      Catch e As Exception
         Console.WriteLine("{0}: Unable to instantiate ar
                           e.GetType().Name, WORD_CLSID)
      End Try
   End Sub
End Module
' The example displays the following output:
'    Instantiated Type object from CLSID {000209FF-0000-0
'    Instantiated Microsoft.Office.Interop.Word.Applicati
```

Remarks

The GetTypeFromCLSID method supports late-bound access to unmanaged COM objects from .NET Framework apps when you know the COM object's class identifier (CLSID). The class identifier for COM classes is defined in the HKEY_CLASSES_ROOT\CLSID key of the registry. You can retrieve the value of the IsCOMObject property to determine whether the type returned by this method is a COM object.

> **Tip**
>
> You can call the **GetTypeFromProgID** method for late-bound access to COM objects whose programmatic identifier (ProgID) you know.

Instantiating an unmanaged COM object from its CLSID is a two-step process:

1. Get a Type object that represents the `__ComObject` that corresponds to the CLSID by calling the GetTypeFromCLSID method.
2. Call the Activator.CreateInstance(Type) method to instantiate the COM object.

Exceptions such as OutOfMemoryException will be thrown when specifying `true` for `throwOnError`, but it will not fail for

unregistered CLSIDs.

Notes to Callers

This method is intended for use when working with COM objects, not with .NET Framework objects. All managed objects, including those that are visible to COM (that is, their ComVisibleAttribute attribute is `true`) have a GUID that is returned by the GUID property. Although the GetTypeFromCLSID(Guid, String, Boolean) method returns a Type object that corresponds to the GUID for a particular managed object, you can't use that Type object to create a type instance by calling the CreateInstance(Type) method, as the following example shows.

```
using System;
using System.Runtime.InteropServices;

[assembly:ComVisible(true)]

// Define two classes, and assign one an explicit GUID.
[GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")]
public class ExplicitGuid
{ }

public class NoExplicitGuid
{ }

public class Example
{
   public static void Main()
   {
      Type explicitType = typeof(ExplicitGuid);
      Guid explicitGuid = explicitType.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type explicitCOM = Type.GetTypeFromCLSID(explicitGu
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid);

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
```

```
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(explicitCO
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }

      Type notExplicit = typeof(NoExplicitGuid);
      Guid notExplicitGuid = notExplicit.GUID;

      // Get type of ExplicitGuid from its GUID.
      Type notExplicitCOM = Type.GetTypeFromCLSID(notExpl
      Console.WriteLine("Created {0} type from CLSID {1}"
                        notExplicitCOM.Name, notExplicitC

      // Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      // Instantiate an ExplicitGuid object.
      try {
         Object obj = Activator.CreateInstance(notExplici
         Console.WriteLine("Instantiated a {0} object", o
      }
      catch (COMException e) {
         Console.WriteLine("COM Exception:\n{0}\n", e.Mes
      }
   }
}
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
```

```
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
open System
open System.Runtime.InteropServices

[<assembly: ComVisible true>]
do ()

// Define two classes, and assign one an explicit GUID.
[<Guid "d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4">]
type ExplicitGuid() = class end

type NoExplicitGuid() = class end

let explicitType = typeof<ExplicitGuid>
let explicitGuid = explicitType.GUID

// Get type of ExplicitGuid from its GUID.
let explicitCOM = Type.GetTypeFromCLSID explicitGuid
printfn $"Created {explicitCOM.Name} type from CLSID {exp

// Compare the two type objects.
printfn $"{explicitType.Name} and {explicitCOM.Name} equa

// Instantiate an ExplicitGuid object.
try
    let obj = Activator.CreateInstance explicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"

let notExplicit = typeof<NoExplicitGuid>
let notExplicitGuid = notExplicit.GUID

// Get type of ExplicitGuid from its GUID.
let notExplicitCOM = Type.GetTypeFromCLSID(notExplicitGui
printfn $"Created {notExplicitCOM.Name} type from CLSID {

// Compare the two type objects.
printfn $"{notExplicit.Name} and {notExplicitCOM.Name} eq

// Instantiate an ExplicitGuid object.
```

```
try
    let obj = Activator.CreateInstance notExplicitCOM
    printfn $"Instantiated a {obj.GetType().Name} object'
with :? COMException as e ->
    printfn $"COM Exception:\n{e.Message}\n"
// The example displays the following output:
//       Created __ComObject type from CLSID d055cba3-1f8
//       ExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
//
//       Created __ComObject type from CLSID 74f03346-a71
//       NoExplicitGuid and __ComObject equal: False
//       COM Exception:
//       Retrieving the COM class factory for component w
//       {74F03346-A718-3516-AC78-F351C7459FFB} failed du
//       80040154 Class not registered
//       (Exception from HRESULT: 0x80040154 (REGDB_E_CLA
```

```
Imports System.Runtime.InteropServices

<Assembly:ComVisible(True)>

' Define two classes, and assign one an explicit GUID.
<GuidAttribute("d055cba3-1f83-4bd7-ba19-e22b1b8ec3c4")>
Public Class ExplicitGuid
End Class

Public Class NoExplicitGuid
End Class

Module Example
   Public Sub Main()
      Dim explicitType As Type = GetType(ExplicitGuid)
      Dim explicitGuid As Guid = explicitType.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim explicitCOM As Type = Type.GetTypeFromCLSID(exp
      Console.WriteLine("Created {0} type from CLSID {1}'
                        explicitCOM.Name, explicitGuid)
```

```
      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        explicitType.Name, explicitCOM.Na
                        explicitType.Equals(explicitCOM))

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(exp
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try

      Dim notExplicit As Type = GetType(NoExplicitGuid)
      Dim notExplicitGuid As Guid = notExplicit.GUID

      ' Get type of ExplicitGuid from its GUID.
      Dim notExplicitCOM As Type = Type.GetTypeFromCLSID(
      Console.WriteLine("Created {0} type from CLSID {1}"
                        notExplicitCOM.Name, notExplicitC

      ' Compare the two type objects.
      Console.WriteLine("{0} and {1} equal: {2}",
                        notExplicit.Name, notExplicitCOM.
                        notExplicit.Equals(notExplicitCOM

      ' Instantiate an ExplicitGuid object.
      Try
         Dim obj As Object = Activator.CreateInstance(not
         Console.WriteLine("Instantiated a {0} object", c
      Catch e As COMException
         Console.WriteLine("COM Exception:{1}{0}{1}", e.M
      End Try
   End Sub
End Module
' The example displays the following output:
'       Created __ComObject type from CLSID d055cba3-1f83
'       ExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
'       {D055CBA3-1F83-4BD7-BA19-E22B1B8EC3C4} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
'
'       Created __ComObject type from CLSID 74f03346-a718
'       NoExplicitGuid and __ComObject equal: False
'       COM Exception:
'       Retrieving the COM class factory for component wi
```

```
'       {74F03346-A718-3516-AC78-F351C7459FFB} failed due
'       80040154 Class not registered
'       (Exception from HRESULT: 0x80040154 (REGDB_E_CLAS
```

Instead, the GetTypeFromCLSID(Guid, String, Boolean) should only be used to retrieve the GUID of an unmanaged COM object, and the resulting Type object that is passed to the CreateInstance(Type) method must represent an unmanaged COM object.

Applies to

▼

Collaborate with us on GitHub

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see our contributor guide.

.NET

.NET feedback

.NET is an open source project. Select a link to provide feedback:

Open a documentation issue

Provide product feedback