

## Re: Array Marshalling different between VB6 & Script to an interop C#

---

*Source:*

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.interop/2006-08/msg00243.html>

---

- *From:* "TDC" <[NOTcarvinSPAM@xxxxxxxxxx](mailto:NOTcarvinSPAM@xxxxxxxxxx)>
  - *Date:* 29 Aug 2006 05:36:32 -0700
- 

I think you need to be declaring a Variant which contains an array or strings, as opposed to an array of strings.

The scripts require all data types to be Variants.

Tom

KirkViehland wrote:

I have a very simple C# library that demonstrates a problem that I am trying to figure out right now. I could post a full example, if this forum allowed posting attachments.

Anyway, like I said, I have a very simple C# class...

```
namespace ScriptMarshalTester
{
    using System;
    using System.Reflection;
    using System.Runtime.InteropServices;

    [ComVisible(true)]
    [Guid("AD07359C-D5D6-4A22-89D7-7DA80D3C4973")]
    [ProgId("ScriptMarshalTester.StringTester")]
    [ClassInterface(ClassInterfaceType.None)]
    public class StringTester
        : ScriptMarshalTester.IStringTester
    {
        public StringTester(){}

        public void TestString
        (
            [In, MarshalAs(UnmanagedType.BStr)]
            string value
        )
        {
            Console.WriteLine( "\r\n" + MethodInfo.GetCurrentMethod().Name +
```

Re: Array Marshalling different between VB6 & Script to an interop C#

```
"\r\n" );
Console.WriteLine( new String('-',20) );
Console.WriteLine( "value=" + value );
}

public void TestStringArray
(
[MarshalAs(UnmanagedType.SafeArray,
SafeArraySubType=VarEnum.VT_BSTR)]
ref string[] values
)
{
Console.WriteLine( "\r\n" + MethodInfo.GetCurrentMethod().Name +
"\r\n" );
Console.WriteLine( new String('-',20) );

for ( int i = 0; i < values.Length; i++ )
Console.WriteLine( "values[" + i.ToString() + "]= " +
values[i]);
}

public void TestStringAndStringArray
(
[In, MarshalAs(UnmanagedType.BStr)]
string value,

[In, Out, MarshalAs(UnmanagedType.SafeArray,
SafeArraySubType=VarEnum.VT_BSTR)]
ref string[] values
)
{
Console.WriteLine( "\r\n" + MethodInfo.GetCurrentMethod().Name +
"\r\n" );
Console.WriteLine( new String('-',20) );

Console.WriteLine( "value=" + value );

for ( int i = 0; i < values.Length; i++ )
Console.WriteLine( "values[" + i.ToString() + "]= " +
values[i]);
}
}
}
```

And its interface definition is as follows...

```
namespace ScriptMarshalTester
{
using System;
using System.Runtime.InteropServices;
```

## Re: Array Marshalling different between VB6 & Script to an interop C#

```
[ComVisible(true)]
[Guid("10F83921-A0EA-4270-8574-5D7577EB72EA")]
[InterfaceType(ComInterfaceType.InterfaceIsIDispatch)]
public interface IStringTester
{
    [DispId(1)]
    void TestString
    (
        [In, MarshalAs(UnmanagedType.BStr)]
        string value
    );

    [DispId(2)]
    void TestStringArray
    (
        [In, MarshalAs(UnmanagedType.SafeArray,
        SafeArraySubType=VarEnum.VT_BSTR)]
        ref string[] values
    );

    [DispId(3)]
    void TestStringAndStringArray
    (
        [In, MarshalAs(UnmanagedType.BStr)]
        string value,

        [In, MarshalAs(UnmanagedType.SafeArray,
        SafeArraySubType=VarEnum.VT_BSTR)]
        ref string[] values
    );
}
}
```

Now, if I compile this and try to execute it with the following VB6 code everything works as expected, \*\* Meaning no exceptions are thrown \*\*. Here is some sample VB6 code. You will notice in this snippet that the original library has been compiled and registered with COM as ScriptMarshalTester. The ProgId for the class that we will be interacting with is ScriptMarshalTester.StringTester. Here is the VB6 subroutine...

```
Private Sub TestStrings()
    Dim tester As ScriptMarshalTester.StringTester
    Dim testValue As String
    Dim testValues() As String

    ReDim testValues(1)

    Set tester = New ScriptMarshalTester.StringTester

    testValue = "Hello World"
    testValues(0) = "Hello"
```

## Re: Array Marshalling different between VB6 & Script to an interop C#

```
testValues(1) = "World"
```

```
Call tester.TestString(testValue)
```

```
Call tester.TestStringArray(testValues())
```

```
Call tester.TestStringAndStringArray(testValue, testValues())
```

```
End Sub
```

At this point, everything behaves as expected. Once we introduce the script client into the mix things start to take a quick downhill. I have 2 script clients which both exhibit the same behavior, a VBScript client and a JScript client. Here is the same method as the TestStrings method above in VBScript...

```
private sub TestStrings()
```

```
dim tester
```

```
dim testValue
```

```
dim testValues()
```

```
ReDim testValues(1)
```

```
set tester = WScript.CreateObject("ScriptMarshalTester.StringTester")
```

```
testValue = "Hello World"
```

```
testValues(0) = "Hello"
```

```
testValues(1) = "World"
```

```
call tester.TestString(testValue)
```

```
call tester.TestStringArray(testValues)
```

```
call tester.TestStringAndStringArray(testValue,testValues)
```

```
end sub
```

And also in the following JScript...

```
function testStrings()
```

```
{
```

```
var tester = new ActiveXObject("ScriptMarshalTester.StringTester");
```

```
var testValue = new String("Hello World");
```

```
var testValues = new Array();
```

```
testValues[0] = "Hello";
```

```
testValues[1] = "World";
```

```
try
```

```
{
```

```
tester.TestString(testValue);
```

```
tester.TestStringArray
```

```
(
```

```
CreateSafeArray(testValues)
```

```
);
```

```
tester.TestStringAndStringArray
```

Re: Array Marshalling different between VB6 & Script to an interop C#

```
(
testValue,
CreateSafeArray(testValues)
);
}
catch(e)
{
WScript.StdOut.WriteLine(e);
WScript.StdOut.WriteLine(e.number & 0xFFFF);
WScript.StdOut.WriteLine(e.description);
}
}

// Converts a JScript Array to a SAFEARRAY
function CreateSafeArray( jsArray )
{
var dictionary = new ActiveXObject("Scripting.Dictionary");

for ( var i = 0; i < jsArray.length; i++ )
dictionary.add( i, jsArray[i] );

return(new VBArray(dictionary.Items()));
}
```

The problem surfaces in that the `tester.TestStringArray` method gets an "Invalid procedure call or argument" while trying to pass the array of strings to the `TestStringArray` method. Likewise, if the same script is coded in JScript and the argument is converted to a `SAFEARRAY` the executing script gets an "Invalid Procedure call or argument" while trying to marshal the array parameter to the managed code.

Unless I have missed something in the documentation, I think that I have declared these interfaces correctly. I know that it is something very simple, but I can't seem to make this work. One requirement however is that the same .Net component should be useable from both VB6 as well as script components.

Any help would be appreciated.

Thanks,  
Kirk A. Viehland