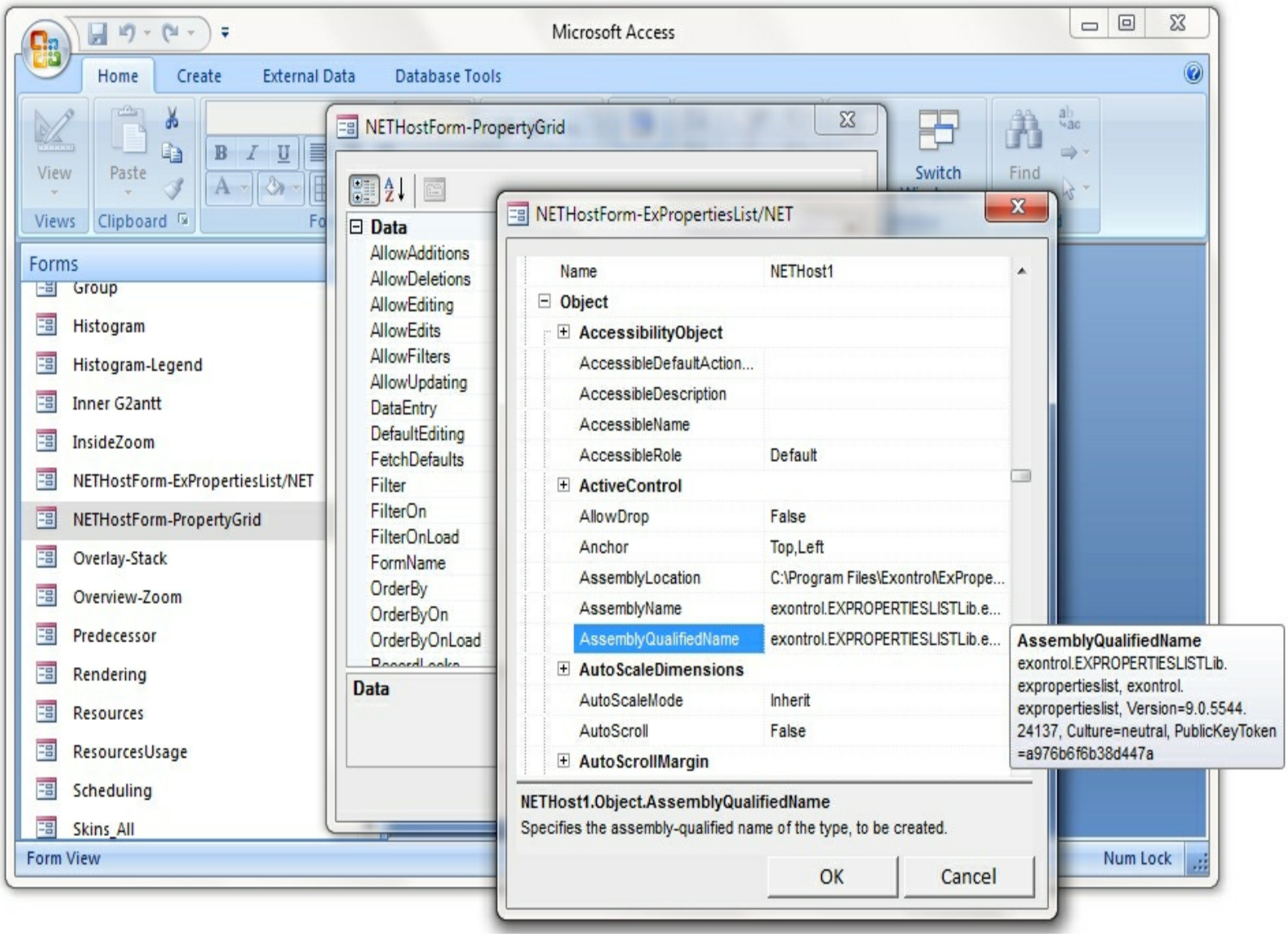
NETHost Component

The Exontrol's NETHost control allows you to use/display any object or Windows Form User Control from /NET framework on your /COM projects / windows / forms / dialogs. The Exontrol's NETHost takes the fully qualified path of the assembly/file or/and the assembly/qualified name of the type, and shows it to your window / form / dialog. For instance, if NETHost.AssemblyQualifiedName property is set to "System.Windows.Forms.MonthCalendar, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" it shows the /NET MonthCalendar component on your window / form / dialog.

Features include:

- Ability to host almost any /NET component on your /COM framework
- Events support, so events of the hosting component are fired through the HostEvent notification
- Handling errors and exceptions of the hosting component at runtime
- X-Script/Template support for /NET objects
- Ability to use/display any /NET component on HTML pages
- Ability to execute code from strings/x-code



How to get support?

To keep your business applications running, you need support you can count on.

Here are few hints what to do when you're stuck on your programming:

- Check out the samples - they are here to provide some quick info on how things should be done
- Check out the how-to questions using the eXHelper tool
- Check out the help - includes documentation for each method, property or event
- Check out if you have the latest version, and if you don't have it send an update request here.
- Submit your problem(question) here.

Don't forget that you can contact our development team if you have ideas or requests for new components, by sending us an e-mail at support@exontrol.com (please include the name of the product in the subject, ex: exgrid) . We're sure our team of developers will try to find a way to make you happy - and us too, since we helped.

Regards,
Exontrol Development Team

https://www.exontrol.com

constants NETHostType

The NETHostType type specifies the type of the control that the control hosts. The IsCreated / Create method returns a NETHostType expression that can be one of the following:

Name	Value	Description
NETHostType_exNETHostNothing	0	The component hosts nothing.
NETHostType_exNETHostControl	1	The component hosts an object of System.Windows.Forms.Control type.
NETHostType_exNETHostObject	2	The component hosts a general object.

constants NETHostVarEnum

The NETHostVarEnum type specifies the VARIANT types. The VtType property specifies the VARIANT type of the object being hold by a NETHostObject or NETObjectTemplate object. The NETHostVarEnum type supports the following values:

Name	Value	Description
NETHostVarEnum_VT_EMPTY	0	Indicates that a value was not specified.
NETHostVarEnum_VT_NULL	1	Indicates a null value, similar to a null value in SQL.
NETHostVarEnum_VT_I2	2	Indicates a short integer.
NETHostVarEnum_VT_I4	3	Indicates a long integer.
NETHostVarEnum_VT_R4	4	Indicates a float value.
NETHostVarEnum_VT_R8	5	Indicates a double value.
NETHostVarEnum_VT_CY	6	Indicates a currency value.
NETHostVarEnum_VT_DATE	7	Indicates a DATE value.
NETHostVarEnum_VT_BSTR	8	Indicates a BSTR string.
NETHostVarEnum_VT_DISPATCH	9	Indicates an IDispatch pointer.
NETHostVarEnum_VT_ERROR	10	Indicates an SCODE.
NETHostVarEnum_VT_BOOL	11	Indicates a Boolean value.
NETHostVarEnum_VT_VARIANT	12	Indicates a VARIANT far pointer.
NETHostVarEnum_VT_UNKNOWN	13	Indicates a IUnknown pointer.
NETHostVarEnum_VT_DECIMAL	14	Indicates a decimal value.
NETHostVarEnum_VT_I1	16	Indicates a char value.
NETHostVarEnum_VT_UI1	17	Indicates a byte.
NETHostVarEnum_VT_UI2	18	Indicates an unsignedshort.
NETHostVarEnum_VT_UI4	19	Indicates an unsignedlong.
NETHostVarEnum_VT_I8	20	Indicates a 64-bit integer.
NETHostVarEnum_VT_UI8	21	Indicates an 64-bit unsigned integer.
NETHostVarEnum_VT_INT	22	Indicates an integer value.
NETHostVarEnum_VT_UINT	23	Indicates an unsigned integer value.
NETHostVarEnum_VT_VOID	24	Indicates a C style void.
NETHostVarEnum_VT_HRESULT	25	Indicates an HRESULT.
NETHostVarEnum_VT_PTR	26	Indicates a pointer type.

NETHostVarEnum_VT_SAFEARRAY	27	Indicates a SAFEARRAY. Not valid in a VARIANT.
NETHostVarEnum_VT_CARRAY	28	Indicates a C style array.
NETHostVarEnum_VT_USERDEFINED	29	Indicates a user defined type.
NETHostVarEnum_VT_LPSTR	30	Indicates a null-terminated string.
NETHostVarEnum_VT_LPWSTR	31	Indicates a wide string terminated by null.
NETHostVarEnum_VT_RECORD	36	Indicates a user defined type.
NETHostVarEnum_VT_FILETIME	64	Indicates a FILETIME value.
NETHostVarEnum_VT_BLOB	65	Indicates length prefixed bytes.
NETHostVarEnum_VT_STREAM	66	Indicates that the name of a stream follows.
NETHostVarEnum_VT_STORAGE	67	Indicates that the name of a storage follows.
NETHostVarEnum_VT_STREAMED_OBJECT	68	Indicates that a stream contains an object.
NETHostVarEnum_VT_STORED_OBJECT	69	Indicates that a storage contains an object.
NETHostVarEnum_VT_BLOB_OBJECT	70	Indicates that a blob contains an object.
NETHostVarEnum_VT_CF	71	Indicates the clipboard format.
NETHostVarEnum_VT_CLSID	72	Indicates a class ID.
NETHostVarEnum_VT_VECTOR	4096	Indicates a simple, counted array.
NETHostVarEnum_VT_ARRAY	8192	Indicates a SAFEARRAY pointer.
NETHostVarEnum_VT_BYREF	16384	Indicates that a value is a reference.

NETHostCtrl object

The Exontrol's NETHost control allows you to use/display any object or Windows Form User Control from /NET framework on your /COM projects / windows / forms / dialogs. The Exontrol's NETHost takes the fully qualified path of the assembly/file or/and the assembly/qualified name of the type, and shows it to your window / form / dialog. For instance, if NETHost.AssemblyQualifiedName property is set to "System.Windows.Forms.MonthCalendar, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" it shows the /NET MonthCalendar component on your window / form / dialog.

Name	Description
AssemblyLocation	Specifies the fully qualified path of the assembly/file to load.
AssemblyName	Specifies the assembly name of the type, to be created (requires AssemblyLocation).
AssemblyQualifiedName	Specifies the assembly-qualified name of the type, to be created.
BackgroundColor	Specifies the hosting's background color.
Create	Creates/Loads the assembly giving fully qualified path of the assembly/file or/and the assembly/qualified name of the type to be created.
Destroy	Destroys the control and unloads the assembly.
Host	Gets the object being hosted.
HostEvents	Specifies the list of events to be handled through the control's Event event, else all events are handled (missing or not set).
hWnd	Indicates the handle to the window (HWND)that hosts the assembly.
IsCreated	Specifies if the assembly is loaded and the control created.
Version	Indicates the version of the NETHost control.

property NETHostCtrl.AssemblyLocation as String

Specifies the fully qualified path of the assembly/file to load.

Type	Description
String	A String expression that specifies the full path or UNC location of the loaded file that contains the manifest/component/assembly.

By default, the AssemblyLocation property is empty. The AssemblyLocation property specifies the full path or UNC location of the loaded file that contains the manifest/component/assembly. The assemblyLocation parameter of the Create method indicates the same value as AssemblyLocation property. In /NET framework, the AssemblyLocation property is similar with the Location property of System.Reflection.Assembly class. The Destroy method unloads the hosting control.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ScrollableControl, System.Windows.Forms,*

Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

The Host property returns the object being hosted by the NETHost control.

If fails, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

property NETHostCtrl.AssemblyName as String

Specifies the assembly name of the type, to be created (requires AssemblyLocation).

Type	Description
String	A string expression that specifies the fully qualified name of the type, including its namespace but not its assembly.

By default, the AssemblyName property is empty. The AssemblyName property specifies the fully qualified name of the type, including its namespace but not its assembly. The assemblyName parameter of the Create method indicates the same value as AssemblyName property. In /NET framework, the AssemblyName property is similar with the FullName property of System.Type class. The Destroy method unloads the hosting control.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ScrollableControl, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*

The Host property returns the object being hosted by the NETHost control.

If fails, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

property NETHostCtrl.AssemblyQualifiedName as String

Specifies the assembly-qualified name of the type, to be created.

Type	Description
String	A String expression that specifies the assembly-qualified name of the type, which includes the name of the assembly from which this type object should be created.

By default, the AssemblyQualifiedName property is empty. The AssemblyQualifiedName property specifies the assembly-qualified name of the type, which includes the name of the assembly from which this type object should be created. In /NET framework, the AssemblyQualifiedName property is similar with the AssemblyQualifiedName property of System.Type class. The Destroy method unloads the hosting control.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ScrollableControl, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*

The Host property returns the object being hosted by the NETHost control.

If fails, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

property NETHostCtrl.BackgroundColor as Long

Specifies the hosting's background color.

Type	Description
Long	A Long expression that specifies the color (RGB color) to be applied to the host's background.

By default, the BackgroundColor property indicates the NETHost container's background color. Use the BackgroundColor property to apply a different background color to the NETHost control. The BackgroundColor property does not change the background color of the hosting control. *To change the background color of the hosting control, you need to consult the hosting control's documentation, and use it in a Template or Item property like in the following samples. Most of the controls provide a BackColor property that change the control's background color, and so that's the property it must be used to change the hosting control's background color.* The BackgroundColor property changes the background color behind the hosting control.

How can I change the control's background color, as BackgroundColor seems to have no effect?

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "BackColor = RGB(240,240,240)"
      With .Item("Nodes.Add(`Root`)")
         Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         Set var_Object = .Item("Expand()")
      End With
   End With
End With
```

VB6

```
With NETHost1
```

```
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "BackColor = RGB(240,240,240)"
      With .Item("Nodes.Add(`Root`)")
         Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         Set var_Object = .Item("Expand()")
      End With
   End With
End With
```

VB.NET

```
Dim var_NETHostObject,var_NETHostObject1,var_Object
With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "BackColor = RGB(240,240,240)"
      With .Item("Nodes.Add(`Root`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         var_Object = .Item("Expand()")
      End With
   End With
End With
```

VB.NET for /COM

```
Dim var_NETHostObject,var_NETHostObject1,var_Object
With AxNETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        .Template = "BackColor = RGB(240,240,240)"
        With .Item("Nodes.Add(`Root`)")
            var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
            var_Object = .Item("Expand()")
        End With
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
>GetHost();
    var_NETHostObject->PutTemplate(L"BackColor = RGB(240,240,240)");
    exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->GetItem(L"Nodes.Add(`Root`)");
        exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->GetItem(L"Nodes.Add(`Child 1`)");
        var_NETHostObject1 = var_NETHostObject1->GetItem(L"Nodes.Add(`Child 2`)");
        ObjectPtr var_Object = ((ObjectPtr)(var_NETHostObject1-
>GetItem(L"Expand()")));
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   var_NETHostObject->Template = L"BackColor = RGB(240,240,240)";
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->get_Item(L"Nodes.Add(`Root`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->get_Item(L"Nodes.Add(`Child 1`)");
      var_NETHostObject1 = var_NETHostObject1->get_Item(L"Nodes.Add(`Child
2`)");
      _tlb::ObjectPtr var_Object = var_NETHostObject1->get_Item(L"Expand()");
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   var_NETHostObject.Template = "BackColor = RGB(240,240,240)";
   exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
      var_NETHostObject1 = var_NETHostObject1["Nodes.Add(`Child 2`)"];
      Object var_Object = (var_NETHostObject1["Expand()"] as Object);
```

JScript/JavaScript

```
<BODY onload="Init()">
```

```
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETHostObject = NETHost1.Host;
      var_NETHostObject.Template = "BackColor = RGB(240,240,240)";
      var var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root`)");
         var var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)");
         var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)");
         var var_Object = var_NETHostObject1.Item("Expand()");
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1">></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
      With .Host
         .Template = "BackColor = RGB(240,240,240)"
         With .Item("Nodes.Add(`Root`)")
```

```
            Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
            Set var_Object = .Item("Expand()")
        End With
    End With
  End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
  var_NETHostObject.Template = "BackColor = RGB(240,240,240)";
  exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root`)"];
    exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
    var_NETHostObject1 = var_NETHostObject1["Nodes.Add(`Child 2`)"];
    Object var_Object = (var_NETHostObject1["Expand()"] as Object);
```

X++ (Dynamics Ax 2009)

```
public void init()
{
  COM com_NETHostObject,com_NETHostObject1,com_Object;
  anytype var_NETHostObject,var_NETHostObject1,var_Object;
  ;

  super();
```

```
exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
      com_NETHostObject.Template("BackColor = RGB(240,240,240)");
      var_NETHostObject1 = com_NETHostObject.Item("Nodes.Add(`Root`)");
com_NETHostObject1 = var_NETHostObject1;
         var_NETHostObject =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 1`)"));
com_NETHostObject = var_NETHostObject;
         var_NETHostObject1 =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 2`)"));
         var_Object = COM::createFromObject(com_NETHostObject1.Item("Expand()"));
com_Object = var_Object;
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'BackColor = RGB(240,240,240)';
      with Item['Nodes.Add(`Root`)'] do
      begin
         var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
         var_NETHostObject1 := Item['Nodes.Add(`Child 2`)'];
         var_Object := (Item['Expand()'] as Object);
      end;
   end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'BackColor = RGB(240,240,240)';
      with Item['Nodes.Add(`Root`)'] do
      begin
         var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
         var_NETHostObject1 := Item['Nodes.Add(`Child 2`)'];
         var_Object := (IUnknown(Item['Expand()']) as _TLB.Object);
      end;
   end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      .Template = "BackColor = RGB(240,240,240)"
      with .Item("Nodes.Add(`Root`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         var_Object = .Item("Expand()")
      endwith
   endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETHostObject,var_NETHostObject1,var_Object
```

```
oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "BackColor = RGB(240,240,240)"
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P
Dim var_NETHostObject1 as P
Dim var_Object as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "BackColor = RGB(240,240,240)"
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

Visual Objects

```
local var_NETHostObject as INETHostObject
local var_NETHostObject1 as INETObjectTemplate
```

```
local var_Object as USUAL

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject:Template := "BackColor = RGB(240,240,240)"
   var_NETHostObject1 := var_NETHostObject:[Item,"Nodes.Add(`Root`)"]
      var_NETHostObject := var_NETHostObject1:[Item,"Nodes.Add(`Child 1`)"]
      var_NETHostObject1 := var_NETHostObject1:[Item,"Nodes.Add(`Child 2`)"]
      var_Object := var_NETHostObject1:[Item,"Expand()"]
```

PowerBuilder

```
OleObject oNETHost,var_NETHostObject,var_NETHostObject1,var_Object

oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "BackColor = RGB(240,240,240)"
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Set ComTemplate of hoNETHostObject to "BackColor = RGB(240,240,240)"
      Variant voNETHostObject1
      Get ComItem of hoNETHostObject "Nodes.Add(`Root`)" to voNETHostObject1
      Handle hoNETHostObject1
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
      Set pvComObject of hoNETHostObject1 to voNETHostObject1
         Variant var_NETHostObject
         Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 1`)" to
var_NETHostObject
         Variant var_NETHostObject1
         Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 2`)" to
var_NETHostObject1
         Variant var_Object
         Get ComItem of hoNETHostObject1 "Expand()" to var_Object
      Send Destroy to hoNETHostObject1
   Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHostObject1,var_NETHostObject,var_NETHostObject1
   LOCAL oNETHost
   LOCAL var_Object
```

```
   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETHostObject := oNETHost:Host()
         oNETHostObject:Template := "BackColor = RGB(240,240,240)"
         oNETHostObject1 := oNETHostObject:Item("Nodes.Add(`Root`)")
            var_NETHostObject := oNETHostObject1:Item("Nodes.Add(`Child 1`)")
            var_NETHostObject1 := oNETHostObject1:Item("Nodes.Add(`Child 2`)")
            var_Object := oNETHostObject1:Item("Expand()")

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

method NETHostCtrl.Create (AssemblyLocation as String, AssemblyName as String)

Creates/Loads the assembly giving fully qualified path of the assembly/file or/and the assembly/qualified name of the type to be created.

Type	Description
AssemblyLocation as String	A String expression that specifies the full path or UNC location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Form*
AssemblyName as String	A String expression that specifies the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.TreeView"*

Return	Description
NETHostType	A NETHostType expression that specifies the type of the object being created, the same as value being returned by the IsCreated property. The return value can be one of the following: • **0**/ NETHostType_exNETHostNothing, The NETHost control hosts nothing. • **1**/ NETHostType_exNETHostControl, The NETHost component hosts an object of System.Windows.Forms.Control type. • **2**/ NETHostType_exNETHostObject, The NETHost component hosts a general object.

The Create method loads the specified file (AssemblyLocation), and creates the giving type (AssemblyName). The assemblyLocation parameter of the Create method indicates the same value as AssemblyLocation property. The assemblyName parameter of the Create method indicates the same value as AssemblyName property. The Create method returns the type of the object being created, the same as IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created. The Host property returns the object being created and hosted by the NETHost control. The Destroy method unloads the hosting control.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the

specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.

- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ScrollableControl, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*

If fails, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

method NETHostCtrl.Destroy ()

Destroys the control and unloads the assembly.

Type	Description

The Destroy method unloads the hosting control. The Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If Destroy method is called, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

property NETHostCtrl.Host as NETHostObject

Gets the object being hosted.

Type	Description
NETHostObject	A NETHostObject object that holds a reference to the hosting control. The Value property of the NETHostObject object specifies the original object being hosted.

By default, the Host property holds nothing. The Host property returns the object being hosted by the NETHost control. Use the AssemblyLocation, AssemblyQualifiedName or Create method to create and host a specified type. **The hosting control's properties or methods must be called using the Item, SetTemplateDef or Template property.** The HostEvent event notifies your application once the hosting control fires an event. The HostEvents property of the NETHost control specifies the list of events that the control should handle.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample:*

"System.Windows.Forms.ScrollableControl, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

The following samples show how you can call hosting control's properties or methods:

- The *Host.Item("GetType().Name").Value* gets the type of the hosting control.
- The *Host.Item("GetType().BaseType().Name").Value* property gets the type of the base of the hosting control.
- The *Host.Item("GetType().IsCOMObject()").Value*, property indicates if the hosting control is /COM object or a/NET assembly.
- The *Host.Item("GetType().GUID()").Value.AsString*, returns the GUID of the hosting control.

property NETHostCtrl.HostEvents as String

Specifies the list of events to be handled through the control's Event event, else all events are handled (missing or not set).

Type	Description
String	A String expression that specifies the list of events to be handled through the control's Event event, else all events are handled (missing or not set). The list of events is separated by any of the following characters: ' ', ',', '.', ':', '\t'

By default, the HostEvents property is empty, which indicates that all events of the hosting event are fired through the NETHost's HostEvent event. The HostEvent event notifies your application once the hosting control (Host) fires an event. The HostEvents property of the NETHost control specifies the list of events that the control should handle. The AsString property of the NETHostEvent object gives a brief description of the event that occurred including the event's name, identifier and its list of arguments. Each control that the NETHost host provides its own events, so for what events the hosting control supports consult its documentation.

Use the following properties to identify/filter the event:

- Name, Indicates the name of the event.
- ID, Indicates the identifier of the event. The ID property may give different values for different versions of hosting control, so you must check for compatibility, so it is not guaranteed that the ID will be unique for any version of the hosting control.
- HostEvents property specifies the list of events to be handled through the control's Event event, else all events are handled (missing or not set).

The following samples filter for "AfterSelect" event of a TreeView control.

VBA (MS Access, Excell...)

```
' HostEvent event - The hosting control fires an event.
Private Sub NETHost1_HostEvent(ByVal Ev As Object)
   With NETHost1
      Debug.Print( Ev.AsString() )
   End With
End Sub

With NETHost1
```

```
    .HostEvents = "AfterSelect"
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        With .Item("Nodes.Add(`Root 1`)")
            Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
            Set var_Object = .Item("Expand()")
        End With
    End With
End With
```

VB6

```
' HostEvent event - The hosting control fires an event.
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
    With NETHost1
        Debug.Print( Ev.AsString() )
    End With
End Sub

With NETHost1
    .HostEvents = "AfterSelect"
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        With .Item("Nodes.Add(`Root 1`)")
            Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
            Set var_Object = .Item("Expand()")
        End With
    End With
End With
```

VB.NET

```
' HostEvent event - The hosting control fires an event.
Private Sub Exnethost1_HostEvent(ByVal sender As System.Object,ByVal Ev As
exontrol.exontrol_NETHost.NETHostEvent) Handles Exnethost1.HostEvent
   With Exnethost1
      Debug.Print( Ev.AsString() )
   End With
End Sub

Dim var_NETHostObject,var_NETHostObject1,var_Object
With Exnethost1
   .HostEvents = "AfterSelect"
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         var_Object = .Item("Expand()")
      End With
   End With
End With
```

VB.NET for /COM

```
' HostEvent event - The hosting control fires an event.
Private Sub AxNETHost1_HostEvent(ByVal sender As System.Object, ByVal e As
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent) Handles
AxNETHost1.HostEvent
   With AxNETHost1
      Debug.Print( e.ev.AsString() )
   End With
End Sub

Dim var_NETHostObject,var_NETHostObject1,var_Object
```

```
With AxNETHost1
   .HostEvents = "AfterSelect"
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
         var_Object = .Item("Expand()")
      End With
   End With
End With
```

C++

```
// HostEvent event - The hosting control fires an event.
void OnHostEventNETHost1(LPDISPATCH Ev)
{
   /*
      Copy and paste the following directives to your header file as
      it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'
      #import <exontrol.NETHost.tlb>
   */
   exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
   OutputDebugStringW( Ev.AsString() );
}

exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1->PutHostEvents(L"AfterSelect");
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form
```

```
spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
>GetHost();
   exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->GetItem(L"Nodes.Add(`Root 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->GetItem(L"Nodes.Add(`Child 1`)");
      var_NETHostObject1 = var_NETHostObject1->GetItem(L"Nodes.Add(`Child 2`)");
      ObjectPtr var_Object = ((ObjectPtr)(var_NETHostObject1-
>GetItem(L"Expand()")));
```

C++ Builder

```
// HostEvent event - The hosting control fires an event.
void __fastcall TForm1::NETHost1HostEvent(TObject
*Sender,Exontrol_nethost_tlb::INETHostEvent *Ev)
{
   OutputDebugString( Ev.AsString() );
}

NETHost1->HostEvents = L"AfterSelect";
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->get_Item(L"Nodes.Add(`Root 1`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->get_Item(L"Nodes.Add(`Child 1`)");
      var_NETHostObject1 = var_NETHostObject1->get_Item(L"Nodes.Add(`Child
2`)");
      _tlb::ObjectPtr var_Object = var_NETHostObject1->get_Item(L"Expand()");
```

C#

```
// HostEvent event - The hosting control fires an event.
private void exnethost1_HostEvent(object
sender,exontrol.exontrol_NETHost.NETHostEvent Ev)
{
   System.Diagnostics.Debug.Print( Ev.ToString() );
}
//this.exnethost1.HostEvent += new
exontrol.exontrol_NETHost.exg2antt.HostEventEventHandler(this.exnethost1_HostEven



exnethost1.HostEvents = "AfterSelect";
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root 1`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
      var_NETHostObject1 = var_NETHostObject1["Nodes.Add(`Child 2`)"];
      Object var_Object = (var_NETHostObject1["Expand()"] as Object);
```

JScript/JavaScript

```
<BODY onload="Init()">
<SCRIPT FOR="NETHost1" EVENT="HostEvent(Ev)" LANGUAGE="JScript">
   alert( Ev.AsString() );
</SCRIPT>

<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
```

```
   NETHost1.HostEvents = "AfterSelect";
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETHostObject = NETHost1.Host;
      var var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)");
         var var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)");
         var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)");
         var var_Object = var_NETHostObject1.Item("Expand()");
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<SCRIPT LANGUAGE="VBScript">
Function NETHost1_HostEvent(Ev)
   With NETHost1
      alert( Ev.AsString() )
   End With
End Function
</SCRIPT>

<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .HostEvents = "AfterSelect"
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
```

```
        With .Host
            With .Item("Nodes.Add(`Root 1`)")
                Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
                Set var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
                Set var_Object = .Item("Expand()")
            End With
        End With
    End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
// HostEvent event - The hosting control fires an event.
private void axNETHost1_HostEvent(object sender,
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent e)
{
    System.Diagnostics.Debug.Print( e.ev.ToString() );
}
//this.axNETHost1.HostEvent += new
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEventHandler(this.axNETHost1_Hos



axNETHost1.HostEvents = "AfterSelect";
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
    exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root 1`)"];
        exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
        var_NETHostObject1 = var_NETHostObject1["Nodes.Add(`Child 2`)"];
        Object var_Object = (var_NETHostObject1["Expand()"] as Object);
```

X++ (Dynamics Ax 2009)

```
// HostEvent event - The hosting control fires an event.
void onEvent_HostEvent(COM _Ev)
{
    ;
    print( _Ev.AsString() );
}

public void init()
{
    COM com_NETHostObject,com_NETHostObject1,com_Object;
    anytype var_NETHostObject,var_NETHostObject1,var_Object;
    ;

    super();

    exnethost1.HostEvents("AfterSelect");

exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

    exnethost1.AssemblyName("System.Windows.Forms.TreeView");
    var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
        var_NETHostObject1 = com_NETHostObject.Item("Nodes.Add(`Root 1`)");
com_NETHostObject1 = var_NETHostObject1;
            var_NETHostObject =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 1`)"));
com_NETHostObject = var_NETHostObject;
            var_NETHostObject1 =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 2`)"));
            var_Object = COM::createFromObject(com_NETHostObject1.Item("Expand()"));
com_Object = var_Object;
}
```

Delphi 8 (.NET only)

```
// HostEvent event - The hosting control fires an event.
procedure TWinForm1.AxNETHost1_HostEvent(sender: System.Object; e:
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent);
begin
   with AxNETHost1 do
   begin
      OutputDebugString( e.ev.AsString() );
   end
end;

with AxNETHost1 do
begin
   HostEvents := 'AfterSelect';
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      with Item['Nodes.Add(`Root 1`)'] do
      begin
         var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
         var_NETHostObject1 := Item['Nodes.Add(`Child 2`)'];
         var_Object := (Item['Expand()'] as Object);
      end;
   end;
end
```

Delphi (standard)

```
// HostEvent event - The hosting control fires an event.
procedure TForm1.NETHost1HostEvent(ASender: TObject; Ev : INETHostEvent);
begin
   with NETHost1 do
   begin
      OutputDebugString( Ev.AsString() );
   end
```

```
end;

with NETHost1 do
begin
   HostEvents := 'AfterSelect';
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      with Item['Nodes.Add(`Root 1`)'] do
      begin
         var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
         var_NETHostObject1 := Item['Nodes.Add(`Child 2`)'];
         var_Object := (IUnknown(Item['Expand()']) as _TLB.Object);
      end;
   end;
end
```

VFP

```
*** HostEvent event - The hosting control fires an event. ***
LPARAMETERS Ev
   with thisform.NETHost1
      DEBUGOUT( Ev.AsString() )
   endwith

with thisform.NETHost1
   .HostEvents = "AfterSelect"
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      with .Item("Nodes.Add(`Root 1`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
```

```
        var_NETHostObject1 = .Item("Nodes.Add(`Child 2`)")
        var_Object = .Item("Expand()")
      endwith
    endwith
endwith
```

dBASE Plus

```
/*
with (this.ACTIVEX1.nativeObject)
   HostEvent = class::nativeObject_HostEvent
endwith
*/
// The hosting control fires an event.
function nativeObject_HostEvent(Ev)
   local oNETHost
   oNETHost = form.Activex1.nativeObject
   ? Str(Ev.AsString())
return

local oNETHost,var_NETHostObject,var_NETHostObject1,var_Object

oNETHost = form.Activex1.nativeObject
oNETHost.HostEvents = "AfterSelect"
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

XBasic (Alpha Five)

```
' The hosting control fires an event.
```

```
function HostEvent as v (Ev as OLE::Exontrol.NETHost::INETHostEvent)
   Dim oNETHost as P
   oNETHost = topparent:CONTROL_ACTIVEX1.activex
   ? Ev.AsString()
end function

Dim oNETHost as P
Dim var_NETHostObject as P
Dim var_NETHostObject1 as P
Dim var_Object as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.HostEvents = "AfterSelect"
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

Visual Objects

```
METHOD OCX_Exontrol1HostEvent(Ev) CLASS MainDialog
   // HostEvent event - The hosting control fires an event.
   OutputDebugString(String2Psz( AsString(Ev.AsString()) ))
RETURN NIL

local var_NETHostObject as INETHostObject
local var_NETHostObject1 as INETObjectTemplate
local var_Object as USUAL

oDCOCX_Exontrol1:HostEvents := "AfterSelect"
oDCOCX_Exontrol1:AssemblyLocation :=
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject1 := var_NETHostObject:[Item,"Nodes.Add(`Root 1`)"]
      var_NETHostObject := var_NETHostObject1:[Item,"Nodes.Add(`Child 1`)"]
      var_NETHostObject1 := var_NETHostObject1:[Item,"Nodes.Add(`Child 2`)"]
      var_Object := var_NETHostObject1:[Item,"Expand()"]
```

PowerBuilder

```
/*begin event HostEvent(oleobject Ev) - The hosting control fires an event.*/
/*
   OleObject oNETHost
   oNETHost = ole_1.Object
   MessageBox("Information",string( String(Ev.AsString()) ))
*/
/*end event HostEvent*/

OleObject oNETHost,var_NETHostObject,var_NETHostObject1,var_Object

oNETHost = ole_1.Object
oNETHost.HostEvents = "AfterSelect"
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject1 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
      var_Object = var_NETHostObject1.Item("Expand()")
```

Visual DataFlex

```
// The hosting control fires an event.
```

```
Procedure OnComHostEvent Variant llEv
   Forward Send OnComHostEvent llEv
   ShowIn llEv
End_Procedure

Procedure OnCreate
   Forward Send OnCreate
   Set ComHostEvents to "AfterSelect"
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Variant voNETHostObject1
      Get ComItem of hoNETHostObject "Nodes.Add(`Root 1`)" to voNETHostObject1
      Handle hoNETHostObject1
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
      Set pvComObject of hoNETHostObject1 to voNETHostObject1
         Variant var_NETHostObject
         Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 1`)" to
var_NETHostObject
         Variant var_NETHostObject1
         Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 2`)" to
var_NETHostObject1
         Variant var_Object
         Get ComItem of hoNETHostObject1 "Expand()" to var_Object
      Send Destroy to hoNETHostObject1
   Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
PROCEDURE OnHostEvent(oNETHost,Ev)
```

```
   DevOut( Transform(Ev,"") )
RETURN

#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHostObject1,var_NETHostObject,var_NETHostObject1
   LOCAL oNETHost
   LOCAL var_Object

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:HostEvent := {|Ev| OnHostEvent(oNETHost,Ev)} /*The hosting control
fires an event.*/

      oNETHost:HostEvents := "AfterSelect"
      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETHostObject := oNETHost:Host()
         oNETHostObject1 := oNETHostObject:Item("Nodes.Add(`Root 1`)")
            var_NETHostObject := oNETHostObject1:Item("Nodes.Add(`Child 1`)")
            var_NETHostObject1 := oNETHostObject1:Item("Nodes.Add(`Child 2`)")
            var_Object := oNETHostObject1:Item("Expand()")
```

```
   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

property NETHostCtrl.hWnd as Long

Indicates the handle to the window (HWND) that hosts the assembly.

Type	Description
Long	A Long expression that indicates the handle to the window (HWND) that hosts the assembly.

The hWnd property returns the handle of the window that displays the NETHost control. The hWnd property does not return the handle of the hosting control. The Microsoft Windows operating environment identifies each form and control in an application by assigning it a handle, or hWnd. The hWnd property is used with Windows API calls. Many Windows operating environment functions require the hWnd of the active window as an argument. The Host property returns the object being created and hosted by the NETHost control. **The hosting control's properties or methods must be called using the Item, SetTemplateDef or Template property.**

The following samples shows how you can get the hwnd/handle of the hosting control. In the following sample, the Item property calls the Handle property of the System.Windows.Forms.Control type.

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Debug.Print( .Host.Item("Handle").AsInt )
End With
```

VB6

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Debug.Print( .Host.Item("Handle").AsInt )
End With
```

VB.NET

```
With Exnethost1
```

```
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    Debug.Print( .Host.Item("Handle").AsInt )
End With
```

VB.NET for /COM

```
With AxNETHost1
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    Debug.Print( .Host.Item("Handle").AsInt )
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

OutputDebugStringW( _bstr_t(spNETHost1->GetHost()->GetItem(L"Handle")-
>GetAsInt()) );
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619
```

```
OutputDebugString( PChar(NETHost1->Host->get_Item(L"Handle")->AsInt) );
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

System.Diagnostics.Debug.Print( exnethost1.Host["Handle"].AsInt.ToString() );
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   alert( NETHost1.Host.Item("Handle").AsInt );
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
```

```
        .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

        alert( .Host.Item("Handle").AsInt )
    End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

System.Diagnostics.Debug.Print( axNETHost1.Host["Handle"].AsInt.ToString() );
```

X++ (Dynamics Ax 2009)

```
public void init()
{
    ;

    super();



exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

    print( exnethost1.Host().Item("Handle").AsInt() );
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
    AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
   OutputDebugString( Host.Item['Handle'].AsInt );
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   OutputDebugString( Host.Item['Handle'].AsInt );
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   DEBUGOUT( .Host.Item("Handle").AsInt )
endwith
```

dBASE Plus

```
local oNETHost

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

? Str(oNETHost.Host.Item("Handle").AsInt)
```

XBasic (Alpha Five)

```
Dim oNETHost as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
```

```
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

? oNETHost.Host.Item("Handle").AsInt
```

Visual Objects

```
oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

OutputDebugString(String2Psz( AsString(oDCOCX_Exontrol1:Host:
[Item,"Handle"]:AsInt) ))
```

PowerBuilder

```
OleObject oNETHost

oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

MessageBox("Information",string( String(oNETHost.Host.Item("Handle").AsInt) ))
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Variant v
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
```

```
  Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
  Set pvComObject of hoNETHostObject to voNETHostObject
    Variant voNETHostObject1
    Get ComItem of hoNETHostObject "Handle" to voNETHostObject1
    Handle hoNETHostObject1
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
    Set pvComObject of hoNETHostObject1 to voNETHostObject1
      Get ComAsInt of hoNETHostObject1 to v
    Send Destroy to hoNETHostObject1
  Send Destroy to hoNETHostObject
  Showln v
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      DevOut( Transform(oNETHost:Host:Item("Handle"):AsInt(),"") )
```

```
   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

property NETHostCtrl.IsCreated as NETHostType

Specifies if the assembly is loaded and the control created.

Type	Description
NETHostType	A NETHostType expression that specifies the type of object being hosted which can be one of the following: • **0**/NETHostType_exNETHostNothing, The NETHost control hosts nothing. • **1**/NETHostType_exNETHostControl, The NETHost component hosts an object of System.Windows.Forms.Control type. • **2**/NETHostType_exNETHostObject, The NETHost component hosts a general object.

By default, the IsCreated property is NETHostType_exNETHostNothing, which indicates that the NETHost control hosts nothing. The IsCreated property specifies the type of the object the NETHost control hosts. The Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type. The Create method returns the same value as IsCreated property. The Destroy method unloads the hosting control.

- The *Host.Item("GetType().Name").Value* gets the type of the hosting control.
- The *Host.Item("GetType().BaseType().Name").Value* property gets the type of the base of the hosting control.
- The *Host.Item("GetType().IsCOMObject()").Value*, property indicates if the hosting control is /COM object or a/NET assembly.
- The *Host.Item("GetType().GUID()").Value.AsString*, returns the GUID of the hosting control.

There are three ways of loading/creating a manifest/component/assembly as listed:

- AssemblyLocation property, loads the first public, browseable control found in the specified location. If there are more public, browseable controls in the assembly, you can use the AssemblyName or AssemblyQualifiedName property to specify the fully qualified name of the type to be hosted.
- AssemblyQualifiedName property, loads and creates the object based on the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ListView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*
- Create(assemblyLocation, assemblyName) method loads the specified file, and creates the giving type.

If succeeded, (the assembly is loaded and the object is created), the

- IsCreated property returns the type of the object being created such as: NETHostType_exNETHostControl, if an object of System.Windows.Forms.Control type was created or NETHostType_exNETHostObject a generic object was created.
- AssemblyLocation property returns the location of the loaded file that contains the manifest/component/assembly. *Sample: "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e*
- AssemblyName property returns the fully qualified name of the type, including its namespace but not its assembly. *Sample: "System.Windows.Forms.ScrollableControl"*
- AssemblyQualifiedName property gets the assembly-qualified name of the type, which includes the name of the assembly from which this type object was loaded. *Sample: "System.Windows.Forms.ScrollableControl, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"*

The Host property returns the object being hosted by the NETHost control.

If fails, the

- IsCreated property returns NETHostType_exNETHostNothing, which indicates no object has been created
- AssemblyLocation, AssemblyName and AssemblyQualifiedName return empty string
- Host property returns nothing.

property NETHostCtrl.Version as String

Indicates the version of the NETHost control.

Type	Description
String	A string expression that indicates the control's version.

The Version property specifies the NETHost control's version. If the Version property includes the DEMO, it indicates that you are running a trial version of the NETHost control. The Version property does not get the hosting control's Version, for that you have to use the AssemblyVersion property of the Host object.

The following samples shows how you can get information about the hosting control, like name, version, ...

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   With .Host
      Debug.Print( .Item("ProductName").Value )
      Debug.Print( .Item("ProductVersion").Value )
      Debug.Print( .Item("CompanyName").Value )
   End With
End With
```

VB6

```
With NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   With .Host
      Debug.Print( .Item("ProductName").Value )
      Debug.Print( .Item("ProductVersion").Value )
      Debug.Print( .Item("CompanyName").Value )
   End With
End With
```

VB.NET

```
With Exnethost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   With .Host
      Debug.Print( .Item("ProductName").Value )
      Debug.Print( .Item("ProductVersion").Value )
      Debug.Print( .Item("CompanyName").Value )
   End With
End With
```

VB.NET for /COM

```
With AxNETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   With .Host
      Debug.Print( .Item("ProductName").Value )
      Debug.Print( .Item("ProductVersion").Value )
      Debug.Print( .Item("CompanyName").Value )
   End With
End With
```

C++

```
/*
   Copy and paste the following directives to your header file as
   it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

   #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1->PutAssemblyQualifiedName(L"System.Windows.Forms.ListBox,
```

```
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1->GetHost();
   OutputDebugStringW( _bstr_t(var_NETHostObject->GetItem(L"ProductName")->GetValue()) );
   OutputDebugStringW( _bstr_t(var_NETHostObject->GetItem(L"ProductVersion")->GetValue()) );
   OutputDebugStringW( _bstr_t(var_NETHostObject->GetItem(L"CompanyName")->GetValue()) );
```

C++ Builder

```
NETHost1->AssemblyQualifiedName = L"System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   OutputDebugString( PChar(var_NETHostObject->get_Item(L"ProductName")->get_Value()) );
   OutputDebugString( PChar(var_NETHostObject->get_Item(L"ProductVersion")->get_Value()) );
   OutputDebugString( PChar(var_NETHostObject->get_Item(L"CompanyName")->get_Value()) );
```

C#

```
exnethost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   System.Diagnostics.Debug.Print(
var_NETHostObject["ProductName"].Value.ToString() );
   System.Diagnostics.Debug.Print(
var_NETHostObject["ProductVersion"].Value.ToString() );
   System.Diagnostics.Debug.Print(
var_NETHostObject["CompanyName"].Value.ToString() );
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089";
   var var_NETHostObject = NETHost1.Host;
      alert( var_NETHostObject.Item("ProductName").Value );
      alert( var_NETHostObject.Item("ProductVersion").Value );
      alert( var_NETHostObject.Item("CompanyName").Value );
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         alert( .Item("ProductName").Value )
         alert( .Item("ProductVersion").Value )
```

```
        alert( .Item("CompanyName").Value )
      End With
    End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
   System.Diagnostics.Debug.Print( var_NETHostObject["ProductName"].Value.ToString() );
   System.Diagnostics.Debug.Print( var_NETHostObject["ProductVersion"].Value.ToString() );
   System.Diagnostics.Debug.Print( var_NETHostObject["CompanyName"].Value.ToString() );
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETHostObject;
   anytype var_NETHostObject;
   ;

   super();

   exnethost1.AssemblyQualifiedName("System.Windows.Forms.ListBox, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089");
   var_NETHostObject = exnethost1.Host(); com_NETHostObject = var_NETHostObject;
      print( com_NETHostObject.Item("ProductName").Value() );
```

```
        print( com_NETHostObject.Item("ProductVersion").Value() );
        print( com_NETHostObject.Item("CompanyName").Value() );
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyQualifiedName := 'System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089';
   with Host do
   begin
      OutputDebugString( Item['ProductName'].Value );
      OutputDebugString( Item['ProductVersion'].Value );
      OutputDebugString( Item['CompanyName'].Value );
   end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyQualifiedName := 'System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089';
   with Host do
   begin
      OutputDebugString( Item['ProductName'].Value );
      OutputDebugString( Item['ProductVersion'].Value );
      OutputDebugString( Item['CompanyName'].Value );
   end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
```

```
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   with .Host
      DEBUGOUT( .Item("ProductName").Value )
      DEBUGOUT( .Item("ProductVersion").Value )
      DEBUGOUT( .Item("CompanyName").Value )
   endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETHostObject

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
var_NETHostObject = oNETHost.Host
   ? Str(var_NETHostObject.Item("ProductName").Value)
   ? Str(var_NETHostObject.Item("ProductVersion").Value)
   ? Str(var_NETHostObject.Item("CompanyName").Value)
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
var_NETHostObject = oNETHost.Host
   ? var_NETHostObject.Item("ProductName").Value
   ? var_NETHostObject.Item("ProductVersion").Value
   ? var_NETHostObject.Item("CompanyName").Value
```

Visual Objects

```
local var_NETHostObject as INETHostObject

oDCOCX_Exontrol1:AssemblyQualifiedName := "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
var_NETHostObject := oDCOCX_Exontrol1:Host
   OutputDebugString(String2Psz( AsString(var_NETHostObject:
[Item,"ProductName"]:Value) ))
   OutputDebugString(String2Psz( AsString(var_NETHostObject:
[Item,"ProductVersion"]:Value) ))
   OutputDebugString(String2Psz( AsString(var_NETHostObject:
[Item,"CompanyName"]:Value) ))
```

PowerBuilder

```
OleObject oNETHost,var_NETHostObject

oNETHost = ole_1.Object
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
var_NETHostObject = oNETHost.Host
   MessageBox("Information",string(
String(var_NETHostObject.Item("ProductName").Value) ))
   MessageBox("Information",string(
String(var_NETHostObject.Item("ProductVersion").Value) ))
   MessageBox("Information",string(
String(var_NETHostObject.Item("CompanyName").Value) ))
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyQualifiedName to "System.Windows.Forms.ListBox,
```

```
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
  Variant voNETHostObject
  Get ComHost to voNETHostObject
  Handle hoNETHostObject
  Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
  Set pvComObject of hoNETHostObject to voNETHostObject
    Variant v
    Variant voNETHostObject1
    Get ComItem of hoNETHostObject "ProductName" to voNETHostObject1
    Handle hoNETHostObject1
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
    Set pvComObject of hoNETHostObject1 to voNETHostObject1
      Get ComValue of hoNETHostObject1 to v
    Send Destroy to hoNETHostObject1
    Showln v
    Variant v1
    Variant voNETHostObject2
    Get ComItem of hoNETHostObject "ProductVersion" to voNETHostObject2
    Handle hoNETHostObject2
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject2
    Set pvComObject of hoNETHostObject2 to voNETHostObject2
      Get ComValue of hoNETHostObject2 to v1
    Send Destroy to hoNETHostObject2
    Showln v1
    Variant v2
    Variant voNETHostObject3
    Get ComItem of hoNETHostObject "CompanyName" to voNETHostObject3
    Handle hoNETHostObject3
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject3
    Set pvComObject of hoNETHostObject3 to voNETHostObject3
      Get ComValue of hoNETHostObject3 to v2
    Send Destroy to hoNETHostObject3
    Showln v2
  Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyQualifiedName := "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      oNETHostObject := oNETHost:Host()
         DevOut( Transform(oNETHostObject:Item("ProductName"):Value(),"") )
         DevOut( Transform(oNETHostObject:Item("ProductVersion"):Value(),"") )
         DevOut( Transform(oNETHostObject:Item("CompanyName"):Value(),"") )

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

NETHostCtrlEvents object

The NETHostEvent object holds information about the event that the Host control fires. The HostEvent event notifies your application once the hosting control (Host) fires an event. The HostEvents property of the NETHost control specifies the list of events that the control should handle.

Name	Description
HostEvent	The hosting control fires an event.

method NETHostCtrlEvents.HostEvent (Ev as NETHostEvent)

The hosting control fires an event.

Type	Description
Ev as NETHostEvent	A NETHostEvent object that holds information about the firing event.

The trial/evaluation version of the control limits firing this event. In other words, using the trial/evaluation version won't fire the event every time it should.

The HostEvent event notifies your application once the hosting control (Host) fires an event. The HostEvents property of the NETHost control specifies the list of events that the control should handle. The AsString property of the NETHostEvent object gives a brief description of the event that occurred including the event's name, identifier and its list of arguments. Each control that the NETHost host provides its own events, so for what events the hosting control supports consult its documentation. The Version property specifies the NETHost control's version, which includes the DEMO if you are running the trial version of the control.

Use the following properties to identify/filter the event:

- Name, Indicates the name of the event.
- ID, Indicates the identifier of the event. The ID property may give different values for different versions of hosting control, so you must check for compatibility, so it is not guaranteed that the ID will be unique for any version of the hosting control.
- HostEvents property specifies the list of events to be handled through the control's Event event, else all events are handled (missing or not set).

Use the following properties to access the arguments of the event:

- AsString, Gives a brief description of the event including its arguments.
- Arguments, gives a NETObjectTemplate object, whose Item or Template properties can be used to access the event's argument using the x-script language.

Syntax for HostEvent event, **/NET** version, on:

C#

```
private void HostEvent(object sender,exontrol.exontrol_NETHost.NETHostEvent
Ev)
{
}
```

VB

```
Private Sub HostEvent(ByVal sender As System.Object,ByVal Ev As
exontrol.exontrol_NETHost.NETHostEvent) Handles HostEvent
```

```
End Sub
```

Syntax for HostEvent event, **/COM** version, on:

C#

```
private void HostEvent(object sender,
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent e)
{
}
```

C++

```
void OnHostEvent(LPDISPATCH Ev)
{
}
```

C++ Builder

```
void __fastcall HostEvent(TObject *Sender,Exontrol_nethost_tlb::INETHostEvent
*Ev)
{
}
```

Delphi

```
procedure HostEvent(ASender: TObject; Ev : INETHostEvent);
begin
end;
```

Delphi 8 (.NET only)

```
procedure HostEvent(sender: System.Object; e:
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent);
begin
end;
```

Powe…

```
begin event HostEvent(oleobject Ev)
end event HostEvent
```

VB.NET

```
Private Sub HostEvent(ByVal sender As System.Object, ByVal e As
Axexontrol_NETHost.INETHostCtrlEvents_HostEventEvent) Handles HostEvent
End Sub
```

VB6

```
Private Sub HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
End Sub
```

VBA

```
Private Sub HostEvent(ByVal Ev As Object)
```

```
End Sub
```

VFP

```
LPARAMETERS Ev
```

Xbas…

```
PROCEDURE OnHostEvent(oNETHost,Ev)
RETURN
```

Syntax for HostEvent event, **/COM** version (others), on:

Java…

```
<SCRIPT EVENT="HostEvent(Ev)" LANGUAGE="JScript">
</SCRIPT>
```

VBSc…

```
<SCRIPT LANGUAGE="VBScript">
Function HostEvent(Ev)
End Function
</SCRIPT>
```

Visual Data…

```
Procedure OnComHostEvent Variant llEv
   Forward Send OnComHostEvent llEv
End_Procedure
```

Visual Objects

```
METHOD OCX_HostEvent(Ev) CLASS MainDialog
RETURN NIL
```

X++

```
void onEvent_HostEvent(COM _Ev)
{
}
```

XBasic

```
function HostEvent as v (Ev as OLE::Exontrol.NETHost::INETHostEvent)
end function
```

dBASE

```
function nativeObject_HostEvent(Ev)
return
```

The following sample displays brief information about firing events:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
```

```
  With NETHost1
    Debug.Print Ev.AsString()
  End With
End Sub
```

The information you get shows as follows:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
NodeMouseHover[15] {Node = TreeNode: Sub-Child 2.1}
Click[42] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
MouseClick[65] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
MouseUp[73] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
MouseMove[72] {Button = None, Clicks = 0, X = 69, Y = 54, Delta = 0, Location = {X=69,Y=54}}
```

Let's explain what data in the AsString representation means:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
```

Having this information about the event, we would know the following:

- Name property that specifies the name of the event, in this case is **MouseMove**
- ID property which specifies the identifier of the event, in this case **72**.
- Arguments such as: **Button**, **Clicks**, **X**, **Y**, **Delta**, **Location.X** and **Location.Y**, ...

The following sample displays each argument of the MouseMove event:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With Ev
    If (.Name = "MouseMove") Then
      Debug.Print .Arguments.Item("Button").AsInt
      Debug.Print .Arguments.Item("Clicks").AsInt
      Debug.Print .Arguments.Item("X").AsInt
      Debug.Print .Arguments.Item("Y").AsInt
      Debug.Print .Arguments.Item("Delta").AsInt
      Debug.Print .Arguments.Item("Location.X").AsInt
```

```
        Debug.Print .Arguments.Item("Location.Y").AsInt
      End If
    End With
End Sub
```

NETHostEvent object

The NETHostEvent object holds information about the event that the Host control fires. The HostEvent event notifies your application once the hosting control (Host) fires an event. The HostEvents property of the NETHost control specifies the list of events that the control should handle. The NETHostEvent object supports the following properties and methods:

Name	Description
Arguments	Gets the arguments of the event.
AsString	Gives a brief description of the event including its arguments.
ID	Indicates the identifier of the event.
Name	Indicates the name of the event.

property NETHostEvent.Arguments as NETObjectTemplate

Gets the arguments of the event.

Type	Description
NETObjectTemplate	A NETObjectTemplate object that holds arguments of the hosting event.

The Arguments property, gives a NETObjectTemplate object, whose Item or Template properties can be used to access the event's argument using the x-script language. The AsString property gives a brief description including name, identifier and arguments of the event.

The following sample displays the number of arguments the event has:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With Ev
    Debug.Print .Arguments.Item("GetType().GetProperties().Length").AsString
  End With
End Sub
```

The following sample displays brief information about firing events:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With NETHost1
    Debug.Print Ev.AsString()
  End With
End Sub
```

The information you get shows as follows:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
NodeMouseHover[15] {Node = TreeNode: Sub-Child 2.1}
Click[42] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
MouseClick[65] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
MouseUp[73] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
MouseMove[72] {Button = None, Clicks = 0, X = 69, Y = 54, Delta = 0, Location =
```

```
{X=69,Y=54}}
```

Let's explain what data in the AsString representation means:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
```

Having this information about the event, we would know the following:

- Name property that specifies the name of the event, in this case is **MouseMove**
- ID property which specifies the identifier of the event, in this case **72**.
- Arguments such as: **Button**, **Clicks**, **X**, **Y**, **Delta**, **Location.X** and **Location.Y**, ...

The following sample displays each argument of the MouseMove event:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With Ev
    If (.Name = "MouseMove") Then
      Debug.Print .Arguments.Item("Button").AsInt
      Debug.Print .Arguments.Item("Clicks").AsInt
      Debug.Print .Arguments.Item("X").AsInt
      Debug.Print .Arguments.Item("Y").AsInt
      Debug.Print .Arguments.Item("Delta").AsInt
      Debug.Print .Arguments.Item("Location.X").AsInt
      Debug.Print .Arguments.Item("Location.Y").AsInt
    End If
  End With
End Sub
```

property NETHostEvent.AsString as String

Gives a brief description of the event including its arguments.

Type	Description
String	A String expression that describes the event being fired including the event's name [event's identifier] { event's arguments }

The AsString property gives a general idea of what data the event contains. The Name property indicates the name of the event. The ID property indicates the identifier of the event. The Arguments property, gives a NETObjectTemplate object, whose Item or Template properties can be used to access the event's argument using the x-script language. The AsString property may returns: "The trial/evaluation version of the control limits firing this event. In other words, using the trial/evaluation version won't fire the event every time it should.", only for not-registered version. The Version property specifies the NETHost control's version, which includes the DEMO if you are running the trial version of the control.

The following sample displays brief information about firing events:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With NETHost1
    Debug.Print Ev.AsString()
  End With
End Sub
```

The information you get shows as follows:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
NodeMouseHover[15] {Node = TreeNode: Sub-Child 2.1}
Click[42] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
MouseClick[65] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
MouseUp[73] {Button = Left, Clicks = 1, X = 69, Y = 53, Delta = 0, Location =
{X=69,Y=53}}
MouseMove[72] {Button = None, Clicks = 0, X = 69, Y = 54, Delta = 0, Location =
{X=69,Y=54}}
```

Let's explain what data in the AsString representation means:

```
MouseMove[72] {Button = Left, Clicks = 0, X = 69, Y = 53, Delta = 0, Location = {X=69,Y=53}}
```

Having this information about the event, we would know the following:

- Name property that specifies the name of the event, in this case is **MouseMove**
- ID property which specifies the identifier of the event, in this case **72**.
- Arguments such as: **Button**, **Clicks**, **X**, **Y**, **Delta**, **Location.X** and **Location.Y**, ...

The following sample displays each argument of the MouseMove event:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With Ev
    If (.Name = "MouseMove") Then
      Debug.Print .Arguments.Item("Button").AsInt
      Debug.Print .Arguments.Item("Clicks").AsInt
      Debug.Print .Arguments.Item("X").AsInt
      Debug.Print .Arguments.Item("Y").AsInt
      Debug.Print .Arguments.Item("Delta").AsInt
      Debug.Print .Arguments.Item("Location.X").AsInt
      Debug.Print .Arguments.Item("Location.Y").AsInt
    End If
  End With
End Sub
```

property NETHostEvent.ID as Long

Indicates the identifier of the event.

Type	Description
Long	A Long expression

The ID property indicates the identifier of the event. The ID property may give different values for different versions of hosting control, so you must check for compatibility, so it is not guaranteed that the ID will be unique for any version of the hosting control. The Name property indicates the name of the event. You can use the Name or ID property to identify a specified event you need to handle in the hosting control. The AsString property gives a brief description including name, identifier and arguments of the event.

The Arguments property, gives a NETObjectTemplate object, whose Item or Template properties can be used to access the event's argument using the x-script language. The AsString property may returns: "The trial/evaluation version of the control limits firing this event. In other words, using the trial/evaluation version won't fire the event every time it should.", only for not-registered version.

The following sample displays the identifier of the firing events:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With NETHost1
    Debug.Print Ev.ID()
  End With
End Sub
```

The information you get shows as follows:

```
72
72
57
58
68
15
67
```

property NETHostEvent.Name as String

Indicates the name of the event.

Type	Description
String	A String expression that specifies the name of the event being fired.

The Name property indicates the name of the event. The ID property indicates the identifier of the event. You can use the Name or ID property to identify a specified event you need to handle in the hosting control. The AsString property gives a brief description including name, identifier and arguments of the event.

The Arguments property, gives a NETObjectTemplate object, whose Item or Template properties can be used to access the event's argument using the x-script language. The AsString property may returns: "The trial/evaluation version of the control limits firing this event. In other words, using the trial/evaluation version won't fire the event every time it should.", only for not-registered version.

The following sample displays the name of the firing events:

```
Private Sub NETHost1_HostEvent(ByVal Ev As exontrol_NETHostCtl.INETHostEvent)
  With NETHost1
    Debug.Print Ev.Name()
  End With
End Sub
```

The information you get shows as follows:

```
MouseDown
MouseCaptureChanged
ItemDrag
MouseMove
MouseMove
NodeMouseHover
Click
MouseClick
```

NETHostObject object

The NETHostObject object holds a .NET Framework object. The NETHostObject type supports the following properties and method:

Name	Description
AsBoolean	Returns a boolean value that represents the current object's value.
AsDate	Returns a date value that represents the current object's value.
AsDouble	Returns a numeric value that represents the current object's value.
AsInt	Returns an integer value that represents the current object's value.
AssemblyVersion	Indicates the version of the assembly being loaded.
AsString	Returns a string that represents the current object's value.
Item	Executes the template and returns the result.
SetTemplateDef	Defines inside variables for the next Template/ExecuteTemplate call.
SetValue	Specifies the value of the object.
Template	Executes the x-script code.
TemplateError	Indicates the error code of the last Template call.
TemplateException	Indicates the detailed information about the exception that occurs.
TemplateResult	Indicates the result of the last Template call.
TemplateThrowError	Specifies whether the execution of the template stops once an error occurs.
Type	Indicates the type of the object's value.
Value	Specifies the value of the object.
VtType	Indicates the type/vartype of the object's value.

property NETHostObject.AsBoolean as Boolean

Returns a boolean value that represents the current object's value.

Type	Description
Boolean	A Boolean expression that specifies the Value converted as boolean.

The AsBoolean property converts the Value to a Boolean expression. If the conversion is not possible, the AsBoolean property returns False. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.AsDate as Date

Returns a date value that represents the current object's value.

Type	Description
Date	A Date/Double expression that specifies the Value converted as date-time.

The AsDate property converts the Value to a DATE-TIME expression. If the conversion is not possible, the AsDate property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.AsDouble as Double

Returns a numeric value that represents the current object's value.

Type	Description
Double	A Double expression that specifies the Value converted as double.

The AsDouble property converts the Value to a double expression. If the conversion is not possible, the AsDouble property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.AsInt as Long

Returns an integer value that represents the current object's value.

Type	Description
Long	A Long expression that specifies the Value converted as long (32-bit integer).

The AsInt property converts the Value to a long expression. If the conversion is not possible, the AsInt property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.AssemblyVersion as String

Indicates the version of the assembly being loaded.

Type	Description
String	A string expression that indicates the hosting control's version.

The AssemblyVersion property indicates the version of the assembly being loaded. The Version property specifies the NETHost control's version.

The following sample shows how you can get the product's version in a different way>

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   Debug.Print( .Host.Item("ProductVersion").Value )
End With
```

VB6

```
With NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   Debug.Print( .Host.Item("ProductVersion").Value )
End With
```

VB.NET

```
With Exnethost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   Debug.Print( .Host.Item("ProductVersion").Value )
End With
```

VB.NET for /COM

```
With AxNETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   Debug.Print( .Host.Item("ProductVersion").Value )
End With
```

C++

```
/*
   Copy and paste the following directives to your header file as
   it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

   #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1->PutAssemblyQualifiedName(L"System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089");
OutputDebugStringW( _bstr_t(spNETHost1->GetHost()-
>GetItem(L"ProductVersion")->GetValue()) );
```

C++ Builder

```
NETHost1->AssemblyQualifiedName = L"System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089";
OutputDebugString( PChar(NETHost1->Host->get_Item(L"ProductVersion")-
>get_Value()) );
```

C#

```
exnethost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
```

```
PublicKeyToken=b77a5c561934e089";
System.Diagnostics.Debug.Print( exnethost1.Host["ProductVersion"].Value.ToString() );
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089";
   alert( NETHost1.Host.Item("ProductVersion").Value );
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      alert( .Host.Item("ProductVersion").Value )
   End With
End Function
</SCRIPT>
```

```
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyQualifiedName = "System.Windows.Forms.ListBox, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089";
System.Diagnostics.Debug.Print( axNETHost1.Host["ProductVersion"].Value.ToString() );
```

X++ (Dynamics Ax 2009)

```
public void init()
{
    ;

    super();

    exnethost1.AssemblyQualifiedName("System.Windows.Forms.ListBox, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089");
    print( exnethost1.Host().Item("ProductVersion").Value() );
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
    AssemblyQualifiedName := 'System.Windows.Forms.ListBox, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089';
    OutputDebugString( Host.Item['ProductVersion'].Value );
end
```

Delphi (standard)

```
with NETHost1 do
```

```
begin
   AssemblyQualifiedName := 'System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089';
   OutputDebugString( Host.Item['ProductVersion'].Value );
end
```

VFP

```
with thisform.NETHost1
   .AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   DEBUGOUT( .Host.Item("ProductVersion").Value )
endwith
```

dBASE Plus

```
local oNETHost

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
? Str(oNETHost.Host.Item("ProductVersion").Value)
```

XBasic (Alpha Five)

```
Dim oNETHost as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
? oNETHost.Host.Item("ProductVersion").Value
```

Visual Objects

```
oDCOCX_Exontrol1:AssemblyQualifiedName := "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
OutputDebugString(String2Psz( AsString(oDCOCX_Exontrol1:Host:
[Item,"ProductVersion"]:Value) ))
```

PowerBuilder

```
OleObject oNETHost

oNETHost = ole_1.Object
oNETHost.AssemblyQualifiedName = "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
MessageBox("Information",string(
String(oNETHost.Host.Item("ProductVersion").Value) ))
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyQualifiedName to "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
   Variant v
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Variant voNETHostObject1
      Get ComItem of hoNETHostObject "ProductVersion" to voNETHostObject1
      Handle hoNETHostObject1
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
      Set pvComObject of hoNETHostObject1 to voNETHostObject1
```

```
        Get ComValue of hoNETHostObject1 to v
      Send Destroy to hoNETHostObject1
    Send Destroy to hoNETHostObject
    Showln v
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyQualifiedName := "System.Windows.Forms.ListBox,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      DevOut( Transform(oNETHost:Host:Item("ProductVersion"):Value(),"") )

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

property NETHostObject.AsString as String

Returns a string that represents the current object's value.

Type	Description
String	A String expression that specifies the Value converted as string.

The AsString property converts the Value to a string expression. If the conversion is not possible, the AsString property returns "" (empty). The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.Item (Template as String) as NETObjectTemplate

Executes the template and returns the result.

Type	Description
Template as String	A String expression that specifies the x-script/template code to be executed.
NETObjectTemplate	A NETObjectTemplate property that holds the result of the last instruction within the Template.

Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. Prior to Template/Item call, you can invoke the SetTemplateDef to define values from your code to Template's code (TemplateDef variables).

in VB/NET under the .NET Framework, you use a code like follows to add nodes to a TreeView control:

```
With TreeView1
  With .Nodes.Add("Root 1")
    .Nodes.Add("Child 1")
    With .Nodes.Add("Child 2")
      .Nodes.Add("Sub-Child 2.1")
      .Nodes.Add("Sub-Child 2.2")
      .Nodes.Add("Sub-Child 2.3")
      .Expand()
    End With
    .Nodes.Add("Child 3")
    .Expand()
  End With
  With .Nodes.Add("Root 2")
    .Nodes.Add("Child 1")
    .Nodes.Add("Child 2")
    .Nodes.Add("Child 3")
  End With
End With
```

while on VB using the NETHost control you should use a code like:

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         .Template = "Nodes.Add(`Child 1`)"
         With .Item("Nodes.Add(`Child 2`)")
            .Template = "Nodes.Add(`Sub-Child 2.1`)"
            .Template = "Nodes.Add(`Sub-Child 2.2`)"
            .Template = "Nodes.Add(`Sub-Child 2.3`)"
            .Template = "Expand()"
         End With
         .Template = "Nodes.Add(`Child 3`)"
         .Template = "Expand()"
      End With
      With .Item("Nodes.Add(`Root 2`)")
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Nodes.Add(`Child 3`)"
      End With
   End With
End With
```

The Template / x-script is composed by lines of instructions. Instructions are separated by "\r\n" (new line characters) or ";" character. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails. The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object.

An x-script instruction/line can be one of the following:

- **Dim variable**[, variable, ...] *declares the variables in the context. Multiple variables are separated by commas. The SetTemplateDef method can declare new variables to be available for the main context. (Sample: Dim h, h1, h2)*

- **variable =** [object.][property/method(arguments).]**property/method(arguments)** *assigns the result of the* ***property/method*** *call to the* ***variable****. (Sample: h = Nodes.Add(`Node`))*
- [object.][property/method(arguments).]**property(arguments) = value** *assigns the* ***value*** *to the* ***property.*** *(Sample: Nodes.Add(`Node`).BackColor = RGB(255,0,0))*
- [object.][property/method(arguments).]**property/method(arguments)** *invokes the* ***property/method****. (Sample: Nodes.Add(`Node`))*
- **{**context **}** *delimits the object's context. The properties/fields or methods called between { and } are related to the last object returned by the property/method prior to { declaration. (Sample: Nodes{Add(`Child 1`);Add(`Child 2`)})*
- ***.*** *delimits the object than its property or method. (Sample: Nodes.Add(`Element`), or Nodes.Add(`Element`) and Nodes{Add(`Element`)} are equivalents)*

where

- ***variable*** *is the name of a variable declared with Dim command or previously defined using the SetTemplateDef method.*
- ***property*** *is the name of a property/field of the current object in the current context.*
- ***method*** *is the name of a method of the current object in the current context.*
- ***arguments*** *include constants and/or variables and/or property/method calls separated by comma character.*
- ***object*** *can be a variable of an Object type,* ***Me*** *or* ***CreateObject*** *call.*

The x-script uses constant expressions as follows:

- *boolean* expression with possible values as ***True*** or ***False****. The True value is equivalent with -1, while False with 0. (Sample: Visible = False)*
- *numeric* expression may starts with **0x** which indicates a hexa decimal representation, else it should starts with digit, or +/- followed by a digit, and . is the decimal separator. *Sample: 13 indicates the integer 13, or* ***12.45*** *indicates the double expression 12,45. (Sample: BackColor = 0xFF0000)*
- *date* expression is delimited by **#** character in the format **#mm/dd/yyyy hh:mm:ss#**. *Sample: #31/12/1971# indicates the December 31, 1971 (Sample: FirstVisibleDate = #1/1/2001#)*
- *string* expression is delimited by **"** or ` characters. If using the ` character, please make sure that it is different than ' which allows adding comments inline. *Sample:* ***"text"*** *or* ***`text`*** *indicates the string text, while the ' text , specifies the comment text. (Sample: Text = "caption")*

Also , the template or x-script code supports general functions as follows:

- **Me** *property indicates the original object, and it is defined as a predefined variable. (Sample: Me.Nodes.Add(`Root 1`))*

- **RGB(**R,G,B**)** *property retrieves an RGB value, where the R, G, B are byte values that indicates the Red Green Blue bytes for the color being specified. (Sample: Nodes.Add(`Root 1`).BackColor = RGB(255,0,0))*
- **LoadPicture(**file**)** *property loads a picture from a file and returns a Picture object required by the picture properties. (Sample: BackgroundImage = LoadPicture(`C:\exontrol\images\auction.gif`)*
- **CreateObject(***assemblyQualifiedName***)** *property creates an instance of the specified type using that type's default constructor. The assemblyQualifiedName indicates the assembly-qualified name of the type to get. See AssemblyQualifiedName. If the type is in the currently executing assembly or in Mscorlib.dll, it is sufficient to supply the type name qualified by its namespace. (Sample: "CreateObject(`System.Windows.Forms.TabPage, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089`){Text = `Page`;UseVisualStyleBackColor = True}")*

The **Template/x-script** syntax in BNF notation is defined like follows:

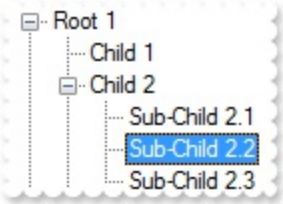
```
<x-script> := <lines>
<lines> := <line>[<eol> <lines>] | <block>
<block> := <call> [<eol>] { [<eol>] <lines> [<eol>] } [<eol>]
<eol> := ";" | "\r\n"
<line> := <dim> | <createobject> | <call> | <set> | <comment>
<dim> := "DIM" <variables>
<variables> := <variable> [, <variables>]
<variable> := "ME" | <identifier>
<createobject> := "CREATEOBJECT(`"<type>"`)"
<call> := <variable> | <property> | <variable>"."<property> | <createobject>"."
<property>
<property> := [<property>"."]<identifier>["("<parameters>")"]
<set> := <call> "=" <value>
<property> := <identifier> | <identifier>"("[<parameters>]")"
<parameters> := <value> [","<parameters>]
<value> := <boolean> | <number> | <color> | <date> | <string> | <createobject> |
<call>
<boolean> := "TRUE" | "FALSE"
<number> := "0X"<hexa> | ["-"]<integer>["."<integer>]
<digit10> := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<digit16> := <digit10> | A | B | C | D | E | F
<integer> := <digit10>[<integer>]
```

```
<hexa> := <digit16>[<hexa>]
<color> := "RGB("<integer>","<integer>","<integer>")"
<date> := "#"<integer>"/"<integer>"/"<integer>" "[<integer>":"<integer>":"
<integer>"]"#"
<string> := '"'<text>'"' | "`"<text>"`"
<comment> := "'"<text>
```

where:

<identifier> indicates an identifier of the variable, property or method, and should start with a letter.
<type> indicates the type the CreateObject function creates, as the assembly-qualified name of the type to create.
<text> any string of characters

The following samples shows how you can use the Item property.



VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         With .Item("Nodes.Add(`Child 2`)")
            Set var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            Set var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            Set var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            Set var_Object = .Item("Expand()")
         End With
         Set var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
         Set var_Object1 = .Item("Expand()")
```

```
        End With
        With .Item("Nodes.Add(`Root 2`)")
            Set var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
            Set var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
        End With
    End With
End With
```

VB6

```
With NETHost1
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        With .Item("Nodes.Add(`Root 1`)")
            Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            With .Item("Nodes.Add(`Child 2`)")
                Set var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
                Set var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
                Set var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
                Set var_Object = .Item("Expand()")
            End With
            Set var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
            Set var_Object1 = .Item("Expand()")
        End With
        With .Item("Nodes.Add(`Root 2`)")
            Set var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
            Set var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
        End With
    End With
End With
```

VB.NET

```
Dim
var_NETHostObject,var_NETHostObject1,var_NETHostObject2,var_NETHostObject3,var

With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         With .Item("Nodes.Add(`Child 2`)")
            var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            var_Object = .Item("Expand()")
         End With
         var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
         var_Object1 = .Item("Expand()")
      End With
      With .Item("Nodes.Add(`Root 2`)")
         var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
         var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
         var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
      End With
   End With
End With
```

VB.NET for /COM

```
Dim
var_NETHostObject,var_NETHostObject1,var_NETHostObject2,var_NETHostObject3,var

With AxNETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        With .Item("Nodes.Add(`Root 1`)")
            var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
            With .Item("Nodes.Add(`Child 2`)")
                var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
                var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
                var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
                var_Object = .Item("Expand()")
            End With
            var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
            var_Object1 = .Item("Expand()")
        End With
        With .Item("Nodes.Add(`Root 2`)")
            var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
            var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
            var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
        End With
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
```

```
>GetHost();
   exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->GetItem(L"Nodes.Add(`Root 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->GetItem(L"Nodes.Add(`Child 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject2 =
var_NETHostObject1->GetItem(L"Nodes.Add(`Child 2`)");
         var_NETHostObject1 = var_NETHostObject2->GetItem(L"Nodes.Add(`Sub-
Child 2.1`)");
         var_NETHostObject2 = var_NETHostObject2->GetItem(L"Nodes.Add(`Sub-
Child 2.2`)");
         exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject3 =
var_NETHostObject2->GetItem(L"Nodes.Add(`Sub-Child 2.3`)");
         ObjectPtr var_Object = ((ObjectPtr)(var_NETHostObject2-
>GetItem(L"Expand()")));
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject4 =
var_NETHostObject1->GetItem(L"Nodes.Add(`Child 3`)");
      ObjectPtr var_Object1 = ((ObjectPtr)(var_NETHostObject1-
>GetItem(L"Expand()")));
   exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject5 =
var_NETHostObject->GetItem(L"Nodes.Add(`Root 2`)");
      var_NETHostObject5 = var_NETHostObject5->GetItem(L"Nodes.Add(`Child 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject6 =
var_NETHostObject5->GetItem(L"Nodes.Add(`Child 2`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject7 =
var_NETHostObject5->GetItem(L"Nodes.Add(`Child 3`)");
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->get_Item(L"Nodes.Add(`Root 1`)");
```

```
    Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject =
var_NETHostObject1->get_Item(L"Nodes.Add(`Child 1`)");
    Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject2 =
var_NETHostObject1->get_Item(L"Nodes.Add(`Child 2`)");
      var_NETHostObject1 = var_NETHostObject2->get_Item(L"Nodes.Add(`Sub-
Child 2.1`)");
      var_NETHostObject2 = var_NETHostObject2->get_Item(L"Nodes.Add(`Sub-
Child 2.2`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject3 =
var_NETHostObject2->get_Item(L"Nodes.Add(`Sub-Child 2.3`)");
      _tlb::ObjectPtr var_Object = var_NETHostObject2->get_Item(L"Expand()");
    Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject4 =
var_NETHostObject1->get_Item(L"Nodes.Add(`Child 3`)");
    _tlb::ObjectPtr var_Object1 = var_NETHostObject1->get_Item(L"Expand()");
  Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject5 =
var_NETHostObject->get_Item(L"Nodes.Add(`Root 2`)");
    var_NETHostObject5 = var_NETHostObject5->get_Item(L"Nodes.Add(`Child
1`)");
    Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject6 =
var_NETHostObject5->get_Item(L"Nodes.Add(`Child 2`)");
    Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject7 =
var_NETHostObject5->get_Item(L"Nodes.Add(`Child 3`)");
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
  exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root 1`)"];
    exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
    exontrol_NETHost.NETHostObject var_NETHostObject2 =
var_NETHostObject1["Nodes.Add(`Child 2`)"];
```

```
        var_NETHostObject1 = var_NETHostObject2["Nodes.Add(`Sub-Child 2.1`)"];
        var_NETHostObject2 = var_NETHostObject2["Nodes.Add(`Sub-Child 2.2`)"];
        exontrol_NETHost.NETHostObject var_NETHostObject3 =
var_NETHostObject2["Nodes.Add(`Sub-Child 2.3`)"];
        Object var_Object = (var_NETHostObject2["Expand()"] as Object);
      exontrol_NETHost.NETHostObject var_NETHostObject4 =
var_NETHostObject1["Nodes.Add(`Child 3`)"];
      Object var_Object1 = (var_NETHostObject1["Expand()"] as Object);
   exontrol_NETHost.NETHostObject var_NETHostObject5 =
var_NETHostObject["Nodes.Add(`Root 2`)"];
      var_NETHostObject5 = var_NETHostObject5["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject6 =
var_NETHostObject5["Nodes.Add(`Child 2`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject7 =
var_NETHostObject5["Nodes.Add(`Child 3`)"];
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETHostObject = NETHost1.Host;
      var var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)");
         var var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)");
         var var_NETHostObject2 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)");
            var_NETHostObject1 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.1`)");
            var_NETHostObject2 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
```

```
2.2`)");
            var var_NETHostObject3 = var_NETHostObject2.Item("Nodes.Add(`Sub-
Child 2.3`)");
            var var_Object = var_NETHostObject2.Item("Expand()");
        var var_NETHostObject4 = var_NETHostObject1.Item("Nodes.Add(`Child 3`)");
        var var_Object1 = var_NETHostObject1.Item("Expand()");
    var var_NETHostObject5 = var_NETHostObject.Item("Nodes.Add(`Root 2`)");
        var_NETHostObject5 = var_NETHostObject5.Item("Nodes.Add(`Child 1`)");
        var var_NETHostObject6 = var_NETHostObject5.Item("Nodes.Add(`Child 2`)");
        var var_NETHostObject7 = var_NETHostObject5.Item("Nodes.Add(`Child 3`)");
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
  With NETHost1
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
      With .Item("Nodes.Add(`Root 1`)")
        Set var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
        With .Item("Nodes.Add(`Child 2`)")
          Set var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
          Set var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
          Set var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
          Set var_Object = .Item("Expand()")
        End With
```

```
            Set var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
            Set var_Object1 = .Item("Expand()")
         End With
         With .Item("Nodes.Add(`Root 2`)")
            Set var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
            Set var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
            Set var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
         End With
      End With
   End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
   exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject["Nodes.Add(`Root 1`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject =
var_NETHostObject1["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject2 =
var_NETHostObject1["Nodes.Add(`Child 2`)"];
         var_NETHostObject1 = var_NETHostObject2["Nodes.Add(`Sub-Child 2.1`)"];
         var_NETHostObject2 = var_NETHostObject2["Nodes.Add(`Sub-Child 2.2`)"];
         exontrol_NETHost.NETHostObject var_NETHostObject3 =
var_NETHostObject2["Nodes.Add(`Sub-Child 2.3`)"];
         Object var_Object = (var_NETHostObject2["Expand()"] as Object);
      exontrol_NETHost.NETHostObject var_NETHostObject4 =
var_NETHostObject1["Nodes.Add(`Child 3`)"];
      Object var_Object1 = (var_NETHostObject1["Expand()"] as Object);
   exontrol_NETHost.NETHostObject var_NETHostObject5 =
```

```
var_NETHostObject["Nodes.Add(`Root 2`)"];
      var_NETHostObject5 = var_NETHostObject5["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject6 =
var_NETHostObject5["Nodes.Add(`Child 2`)"];
      exontrol_NETHost.NETHostObject var_NETHostObject7 =
var_NETHostObject5["Nodes.Add(`Child 3`)"];
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM
com_NETHostObject,com_NETHostObject1,com_NETHostObject2,com_NETHostObject

   anytype
var_NETHostObject,var_NETHostObject1,var_NETHostObject2,var_NETHostObject3,var

   ;

   super();


exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
      var_NETHostObject1 = com_NETHostObject.Item("Nodes.Add(`Root 1`)");
com_NETHostObject1 = var_NETHostObject1;
         var_NETHostObject =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 1`)"));
com_NETHostObject = var_NETHostObject;
         var_NETHostObject2 = com_NETHostObject1.Item("Nodes.Add(`Child 2`)");
com_NETHostObject2 = var_NETHostObject2;
            var_NETHostObject1 =
COM::createFromObject(com_NETHostObject2.Item("Nodes.Add(`Sub-Child 2.1`)"));
```

```
            var_NETHostObject2 =
COM::createFromObject(com_NETHostObject2.Item("Nodes.Add(`Sub-Child 2.2`)"));
            var_NETHostObject3 =
COM::createFromObject(com_NETHostObject2.Item("Nodes.Add(`Sub-Child 2.3`)"));
com_NETHostObject3 = var_NETHostObject3;
            var_Object =
COM::createFromObject(com_NETHostObject2.Item("Expand()")); com_Object =
var_Object;
        var_NETHostObject4 =
COM::createFromObject(com_NETHostObject1.Item("Nodes.Add(`Child 3`)"));
com_NETHostObject4 = var_NETHostObject4;
        var_Object1 =
COM::createFromObject(com_NETHostObject1.Item("Expand()")); com_Object1 =
var_Object1;
      var_NETHostObject5 = com_NETHostObject.Item("Nodes.Add(`Root 2`)");
com_NETHostObject5 = var_NETHostObject5;
        var_NETHostObject5 =
COM::createFromObject(com_NETHostObject5.Item("Nodes.Add(`Child 1`)"));
        var_NETHostObject6 =
COM::createFromObject(com_NETHostObject5.Item("Nodes.Add(`Child 2`)"));
com_NETHostObject6 = var_NETHostObject6;
        var_NETHostObject7 =
COM::createFromObject(com_NETHostObject5.Item("Nodes.Add(`Child 3`)"));
com_NETHostObject7 = var_NETHostObject7;
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      with Item['Nodes.Add(`Root 1`)'] do
```

```
      begin
        var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
        with Item['Nodes.Add(`Child 2`)'] do
        begin
          var_NETHostObject1 := Item['Nodes.Add(`Sub-Child 2.1`)'];
          var_NETHostObject2 := Item['Nodes.Add(`Sub-Child 2.2`)'];
          var_NETHostObject3 := Item['Nodes.Add(`Sub-Child 2.3`)'];
          var_Object := (Item['Expand()'] as Object);
        end;
        var_NETHostObject4 := Item['Nodes.Add(`Child 3`)'];
        var_Object1 := (Item['Expand()'] as Object);
      end;
      with Item['Nodes.Add(`Root 2`)'] do
      begin
        var_NETHostObject5 := Item['Nodes.Add(`Child 1`)'];
        var_NETHostObject6 := Item['Nodes.Add(`Child 2`)'];
        var_NETHostObject7 := Item['Nodes.Add(`Child 3`)'];
      end;
    end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
  AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    with Item['Nodes.Add(`Root 1`)'] do
    begin
      var_NETHostObject := Item['Nodes.Add(`Child 1`)'];
      with Item['Nodes.Add(`Child 2`)'] do
      begin
        var_NETHostObject1 := Item['Nodes.Add(`Sub-Child 2.1`)'];
```

```
            var_NETHostObject2 := Item['Nodes.Add(`Sub-Child 2.2`)'];
            var_NETHostObject3 := Item['Nodes.Add(`Sub-Child 2.3`)'];
            var_Object := (IUnknown(Item['Expand()']) as _TLB.Object);
         end;
         var_NETHostObject4 := Item['Nodes.Add(`Child 3`)'];
         var_Object1 := (IUnknown(Item['Expand()']) as _TLB.Object);
      end;
      with Item['Nodes.Add(`Root 2`)'] do
      begin
         var_NETHostObject5 := Item['Nodes.Add(`Child 1`)'];
         var_NETHostObject6 := Item['Nodes.Add(`Child 2`)'];
         var_NETHostObject7 := Item['Nodes.Add(`Child 3`)'];
      end;
   end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      with .Item("Nodes.Add(`Root 1`)")
         var_NETHostObject = .Item("Nodes.Add(`Child 1`)")
         with .Item("Nodes.Add(`Child 2`)")
            var_NETHostObject1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            var_NETHostObject2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            var_NETHostObject3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            var_Object = .Item("Expand()")
         endwith
         var_NETHostObject4 = .Item("Nodes.Add(`Child 3`)")
         var_Object1 = .Item("Expand()")
      endwith
      with .Item("Nodes.Add(`Root 2`)")
         var_NETHostObject5 = .Item("Nodes.Add(`Child 1`)")
```

```
        var_NETHostObject6 = .Item("Nodes.Add(`Child 2`)")
        var_NETHostObject7 = .Item("Nodes.Add(`Child 3`)")
      endwith
    endwith
endwith
```

dBASE Plus

```
local
oNETHost,var_NETHostObject,var_NETHostObject1,var_NETHostObject2,var_NETHostO



oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject2 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
         var_NETHostObject1 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.1`)")
         var_NETHostObject2 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.2`)")
         var_NETHostObject3 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.3`)")
         var_Object = var_NETHostObject2.Item("Expand()")
      var_NETHostObject4 = var_NETHostObject1.Item("Nodes.Add(`Child 3`)")
      var_Object1 = var_NETHostObject1.Item("Expand()")
   var_NETHostObject5 = var_NETHostObject.Item("Nodes.Add(`Root 2`)")
      var_NETHostObject5 = var_NETHostObject5.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject6 = var_NETHostObject5.Item("Nodes.Add(`Child 2`)")
      var_NETHostObject7 = var_NETHostObject5.Item("Nodes.Add(`Child 3`)")
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P
Dim var_NETHostObject1 as P
Dim var_NETHostObject2 as P
Dim var_NETHostObject3 as P
Dim var_NETHostObject4 as P
Dim var_NETHostObject5 as P
Dim var_NETHostObject6 as P
Dim var_NETHostObject7 as P
Dim var_Object as P
Dim var_Object1 as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject2 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
         var_NETHostObject1 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.1`)")
         var_NETHostObject2 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.2`)")
         var_NETHostObject3 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.3`)")
         var_Object = var_NETHostObject2.Item("Expand()")
      var_NETHostObject4 = var_NETHostObject1.Item("Nodes.Add(`Child 3`)")
      var_Object1 = var_NETHostObject1.Item("Expand()")
   var_NETHostObject5 = var_NETHostObject.Item("Nodes.Add(`Root 2`)")
      var_NETHostObject5 = var_NETHostObject5.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject6 = var_NETHostObject5.Item("Nodes.Add(`Child 2`)")
      var_NETHostObject7 = var_NETHostObject5.Item("Nodes.Add(`Child 3`)")
```

Visual Objects

```
local var_NETHostObject as INETHostObject
local
var_NETHostObject1,var_NETHostObject2,var_NETHostObject3,var_NETHostObject4,va
 as INETObjectTemplate
local var_Object,var_Object1 as USUAL

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject1 := var_NETHostObject:[Item,"Nodes.Add(`Root 1`)"]
      var_NETHostObject := var_NETHostObject1:[Item,"Nodes.Add(`Child 1`)"]
      var_NETHostObject2 := var_NETHostObject1:[Item,"Nodes.Add(`Child 2`)"]
         var_NETHostObject1 := var_NETHostObject2:[Item,"Nodes.Add(`Sub-Child
2.1`)"]
         var_NETHostObject2 := var_NETHostObject2:[Item,"Nodes.Add(`Sub-Child
2.2`)"]
         var_NETHostObject3 := var_NETHostObject2:[Item,"Nodes.Add(`Sub-Child
2.3`)"]
         var_Object := var_NETHostObject2:[Item,"Expand()"]
      var_NETHostObject4 := var_NETHostObject1:[Item,"Nodes.Add(`Child 3`)"]
      var_Object1 := var_NETHostObject1:[Item,"Expand()"]
   var_NETHostObject5 := var_NETHostObject:[Item,"Nodes.Add(`Root 2`)"]
      var_NETHostObject5 := var_NETHostObject5:[Item,"Nodes.Add(`Child 1`)"]
      var_NETHostObject6 := var_NETHostObject5:[Item,"Nodes.Add(`Child 2`)"]
      var_NETHostObject7 := var_NETHostObject5:[Item,"Nodes.Add(`Child 3`)"]
```

PowerBuilder

```
OleObject
oNETHost,var_NETHostObject,var_NETHostObject1,var_NETHostObject2,var_NETHostO



oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject1 = var_NETHostObject.Item("Nodes.Add(`Root 1`)")
      var_NETHostObject = var_NETHostObject1.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject2 = var_NETHostObject1.Item("Nodes.Add(`Child 2`)")
         var_NETHostObject1 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.1`)")
         var_NETHostObject2 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.2`)")
         var_NETHostObject3 = var_NETHostObject2.Item("Nodes.Add(`Sub-Child
2.3`)")
         var_Object = var_NETHostObject2.Item("Expand()")
      var_NETHostObject4 = var_NETHostObject1.Item("Nodes.Add(`Child 3`)")
      var_Object1 = var_NETHostObject1.Item("Expand()")
   var_NETHostObject5 = var_NETHostObject.Item("Nodes.Add(`Root 2`)")
      var_NETHostObject5 = var_NETHostObject5.Item("Nodes.Add(`Child 1`)")
      var_NETHostObject6 = var_NETHostObject5.Item("Nodes.Add(`Child 2`)")
      var_NETHostObject7 = var_NETHostObject5.Item("Nodes.Add(`Child 3`)")
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Variant voNETHostObject1
      Get ComItem of hoNETHostObject "Nodes.Add(`Root 1`)" to voNETHostObject1
```

```
    Handle hoNETHostObject1
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
    Set pvComObject of hoNETHostObject1 to voNETHostObject1
      Variant var_NETHostObject
      Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 1`)" to
var_NETHostObject
      Variant voNETHostObject2
      Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 2`)" to
voNETHostObject2
      Handle hoNETHostObject2
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject2
      Set pvComObject of hoNETHostObject2 to voNETHostObject2
        Variant var_NETHostObject1
        Get ComItem of hoNETHostObject2 "Nodes.Add(`Sub-Child 2.1`)" to
var_NETHostObject1
        Variant var_NETHostObject2
        Get ComItem of hoNETHostObject2 "Nodes.Add(`Sub-Child 2.2`)" to
var_NETHostObject2
        Variant var_NETHostObject3
        Get ComItem of hoNETHostObject2 "Nodes.Add(`Sub-Child 2.3`)" to
var_NETHostObject3
        Variant var_Object
        Get ComItem of hoNETHostObject2 "Expand()" to var_Object
      Send Destroy to hoNETHostObject2
      Variant var_NETHostObject4
      Get ComItem of hoNETHostObject1 "Nodes.Add(`Child 3`)" to
var_NETHostObject4
      Variant var_Object1
      Get ComItem of hoNETHostObject1 "Expand()" to var_Object1
    Send Destroy to hoNETHostObject1
    Variant voNETHostObject3
    Get ComItem of hoNETHostObject "Nodes.Add(`Root 2`)" to voNETHostObject3
    Handle hoNETHostObject3
    Get Create (RefClass(cComNETHostObject)) to hoNETHostObject3
    Set pvComObject of hoNETHostObject3 to voNETHostObject3
      Variant var_NETHostObject5
      Get ComItem of hoNETHostObject3 "Nodes.Add(`Child 1`)" to
```

```
var_NETHostObject5
        Variant var_NETHostObject6
        Get ComItem of hoNETHostObject3 "Nodes.Add(`Child 2`)" to
var_NETHostObject6
        Variant var_NETHostObject7
        Get ComItem of hoNETHostObject3 "Nodes.Add(`Child 3`)" to
var_NETHostObject7
      Send Destroy to hoNETHostObject3
   Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL
oNETHostObject1,oNETHostObject2,oNETHostObject3,var_NETHostObject,var_NETHo

   LOCAL oNETHost
   LOCAL var_Object,var_Object1

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETHostObject := oNETHost:Host()
         oNETHostObject1 := oNETHostObject:Item("Nodes.Add(`Root 1`)")
            var_NETHostObject := oNETHostObject1:Item("Nodes.Add(`Child 1`)")
            oNETHostObject2 := oNETHostObject1:Item("Nodes.Add(`Child 2`)")
               var_NETHostObject1 := oNETHostObject2:Item("Nodes.Add(`Sub-Child
2.1`)")
               var_NETHostObject2 := oNETHostObject2:Item("Nodes.Add(`Sub-Child
2.2`)")
               var_NETHostObject3 := oNETHostObject2:Item("Nodes.Add(`Sub-Child
2.3`)")
               var_Object := oNETHostObject2:Item("Expand()")
            var_NETHostObject4 := oNETHostObject1:Item("Nodes.Add(`Child 3`)")
            var_Object1 := oNETHostObject1:Item("Expand()")
         oNETHostObject3 := oNETHostObject:Item("Nodes.Add(`Root 2`)")
            var_NETHostObject5 := oNETHostObject3:Item("Nodes.Add(`Child 1`)")
            var_NETHostObject6 := oNETHostObject3:Item("Nodes.Add(`Child 2`)")
            var_NETHostObject7 := oNETHostObject3:Item("Nodes.Add(`Child 3`)")

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

method NETHostObject.SetTemplateDef (Value as Variant)

Defines inside variables for the next Template/ExecuteTemplate call.

Type	Description
Value as Variant	If calling the first time, A String expression that indicates the DIM command to define the variables that follows, or a VARIANT expression that defines the value of the variable in the order as they were defined.

The SetTemplateDef method was provided to let you use values/objects inside the next Template/Item call. For instance, let's say you have a date field in your form, and once the user fills it, you want a /NET Frameworks MonthCalendar object to select it. In order to do that you have to call a code like:

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

This sample defines the variable x to be 1/1/2001 for the Template call, so the SelectionStart will be set on 1/1/2001.

The call of SetTemplateDef method consists in:

- **First** call of SetTemplateDef method should be on a form of SetTemplateDef("Dim variable[,variable,...]"). This defines the name of the variable that follow to be defined.
- **Next** calls, must be exactly the same as with the number of variables you defined, which will define the variable one by one. For instance, if your first call was SetTemplateDef("Dim h1,h2,h3"), it means that the next-three calls of SetTemplateDef defines the variable h1, h2 and h3

Once you defined the variables, they will be available for the next calls of Template/Item properties.

The following samples shows how you can define the x variable to be set on the SelectionStart property of the MonthCalendar object:

VBA (MS Access, Excell...)

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

VB6

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

VB.NET

```
With Exnethost1
   .BackgroundColor = 16777215

.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56

   With .Host
```

```
        .SetTemplateDef("Dim x")
        .SetTemplateDef(#1/1/2001#)
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    End With
End With
```

VB.NET for /COM

```
With AxNETHost1
    .BackgroundColor = 16777215

.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56

    With .Host
        .SetTemplateDef("Dim x")
        .SetTemplateDef(#1/1/2001#)
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1->PutBackgroundColor(16777215);
spNETHost1-
>Create(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77

exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
>GetHost();
    var_NETHostObject->SetTemplateDef("Dim x");
```

```
   var_NETHostObject->SetTemplateDef(COleDateTime(2001,1,1,0,00,00).operator
DATE());
   var_NETHostObject->PutTemplate(L"MaxSelectionCount = 1;SelectionStart = x");
```

C++ Builder

```
NETHost1->BackgroundColor = 16777215;
NETHost1-
>Create(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77

Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   var_NETHostObject->SetTemplateDef(TVariant("Dim x"));
   var_NETHostObject->SetTemplateDef(TVariant(TDateTime(2001,1,1).operator
double()));
   var_NETHostObject->Template = L"MaxSelectionCount = 1;SelectionStart = x";
```

C#

```
exnethost1.BackgroundColor = 16777215;
exnethost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0

exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   var_NETHostObject.SetTemplateDef("Dim x");

var_NETHostObject.SetTemplateDef(Convert.ToDateTime("1/1/2001",System.Globalizat
US")));
   var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
```

```
function Init()
{
   NETHost1.BackgroundColor = 16777215;

NETHost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.(

   var var_NETHostObject = NETHost1.Host;
      var_NETHostObject.SetTemplateDef("Dim x");
      var_NETHostObject.SetTemplateDef("1/1/2001");
      var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .BackgroundColor = 16777215
      .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      With .Host
         .SetTemplateDef "Dim x"
         .SetTemplateDef #1/1/2001#
         .Template = "MaxSelectionCount = 1;SelectionStart = x"
      End With
   End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.BackgroundColor = 16777215;
axNETHost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.

exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
   var_NETHostObject.SetTemplateDef("Dim x");

var_NETHostObject.SetTemplateDef(Convert.ToDateTime("1/1/2001",System.Globalizat
US")));
   var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETHostObject;
   anytype var_NETHostObject;
   ;

   super();

   exnethost1.BackgroundColor(16777215);

exnethost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0

   var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
      com_NETHostObject.SetTemplateDef("Dim x");

com_NETHostObject.SetTemplateDef(COMVariant::createFromDate(str2Date("1/1/2001

      com_NETHostObject.Template("MaxSelectionCount = 1;SelectionStart = x");
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   BackgroundColor := 16777215;

Create('C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561

   with Host do
   begin
      SetTemplateDef('Dim x');
      SetTemplateDef('1/1/2001');
      Template := 'MaxSelectionCount = 1;SelectionStart = x';
   end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
   BackgroundColor := 16777215;

Create('C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561

   with Host do
   begin
      SetTemplateDef('Dim x');
      SetTemplateDef('1/1/2001');
      Template := 'MaxSelectionCount = 1;SelectionStart = x';
   end;
end
```

VFP

```
with thisform.NETHost1
   .BackgroundColor = 16777215

.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56
```

```
    with .Host
        .SetTemplateDef("Dim x")
        .SetTemplateDef({^2001-1-1})
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETHostObject

oNETHost = form.Activex1.nativeObject
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETHostObject = oNETHost.Host
    var_NETHostObject.SetTemplateDef("Dim x")
    var_NETHostObject.SetTemplateDef("01/01/2001")
    var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETHostObject = oNETHost.Host
    var_NETHostObject.SetTemplateDef("Dim x")
    var_NETHostObject.SetTemplateDef({01/01/2001})
    var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

Visual Objects

```
local var_NETHostObject as INETHostObject

oDCOCX_Exontrol1:BackgroundColor := 16777215
oDCOCX_Exontrol1:Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms

var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject:SetTemplateDef("Dim x")
   var_NETHostObject:SetTemplateDef(SToD("20010101"))
   var_NETHostObject:Template := "MaxSelectionCount = 1;SelectionStart = x"
```

PowerBuilder

```
OleObject oNETHost,var_NETHostObject

oNETHost = ole_1.Object
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETHostObject = oNETHost.Host
   var_NETHostObject.SetTemplateDef("Dim x")
   var_NETHostObject.SetTemplateDef(2001-01-01)
   var_NETHostObject.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComBackgroundColor to 16777215
   Get ComCreate
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
 "System.Windows.Forms.MonthCalendar" to Nothing
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
```

```
    Send ComSetTemplateDef of hoNETHostObject "Dim x"
    Send ComSetTemplateDef of hoNETHostObject "1/1/2001"
    Set ComTemplate of hoNETHostObject to "MaxSelectionCount =
1;SelectionStart = x"
  Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:BackgroundColor := 16777215

oNETHost:Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

      oNETHostObject := oNETHost:Host()
         oNETHostObject:SetTemplateDef("Dim x")
         oNETHostObject:SetTemplateDef("01/01/2001")
         oNETHostObject:Template := "MaxSelectionCount = 1;SelectionStart = x"
```

```
   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

method NETHostObject.SetValue (Value as Variant)

Specifies the value of the object.

Type	Description
Value as Variant	A VARIANT expression that specifies the new value to be assigned to the NETHostObject object.

Use the SetValue property to change the object being hosted by the current NETHostObject object. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.Template as String

Executes the x-script code.

Type	Description
String	A String expression that specifies the x-script/template code to be executed.

Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. Prior to Template/Item call, you can invoke the SetTemplateDef to define values from your code to Template's code (TemplateDef variables).

in VB/NET under the .NET Framework, you use a code like follows to add nodes to a TreeView control:

```
With TreeView1
   With .Nodes.Add("Root 1")
      .Nodes.Add("Child 1")
      With .Nodes.Add("Child 2")
         .Nodes.Add("Sub-Child 2.1")
         .Nodes.Add("Sub-Child 2.2")
         .Nodes.Add("Sub-Child 2.3")
         .Expand()
      End With
      .Nodes.Add("Child 3")
      .Expand()
   End With
   With .Nodes.Add("Root 2")
      .Nodes.Add("Child 1")
      .Nodes.Add("Child 2")
      .Nodes.Add("Child 3")
   End With
End With
```

while on VB using the NETHost control you should use a code like:

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         .Template = "Nodes.Add(`Child 1`)"
         With .Item("Nodes.Add(`Child 2`)")
            .Template = "Nodes.Add(`Sub-Child 2.1`)"
            .Template = "Nodes.Add(`Sub-Child 2.2`)"
            .Template = "Nodes.Add(`Sub-Child 2.3`)"
            .Template = "Expand()"
         End With
         .Template = "Nodes.Add(`Child 3`)"
         .Template = "Expand()"
      End With
      With .Item("Nodes.Add(`Root 2`)")
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Nodes.Add(`Child 3`)"
      End With
   End With
End With
```

The **Template/x-script** syntax in BNF notation is defined like follows:

```
<x-script> := <lines>
<lines> := <line>[<eol> <lines>] | <block>
<block> := <call> [<eol>] { [<eol>] <lines> [<eol>] } [<eol>]
<eol> := ";" | "\r\n"
<line> := <dim> | <createobject> | <call> | <set> | <comment>
<dim> := "DIM" <variables>
<variables> := <variable> [, <variables>]
<variable> := "ME" | <identifier>
<createobject> := "CREATEOBJECT(`"<type>"`)"
<call> := <variable> | <property> | <variable>"."<property> | <createobject>"."
```

```
<property>
<property> := [<property>"."]<identifier>["("<parameters>")"]
<set> := <call> "=" <value>
<property> := <identifier> | <identifier>"("[<parameters>]")"
<parameters> := <value> [","<parameters>]
<value> := <boolean> | <number> | <color> | <date> | <string> | <createobject> |
<call>
<boolean> := "TRUE" | "FALSE"
<number> := "0X"<hexa> | ["-"]<integer>["."<integer>]
<digit10> := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<digit16> := <digit10> | A | B | C | D | E | F
<integer> := <digit10>[<integer>]
<hexa> := <digit16>[<hexa>]
<color> := "RGB("<integer>","<integer>","<integer>")"
<date> := "#"<integer>"/"<integer>"/"<integer>" "[<integer>":"<integer>":"
<integer>"]"#"
<string> := '"'<text>'"' | "`"<text>"`"
<comment> := "'"<text>
```

where:

<identifier> indicates an identifier of the variable, property or method, and should start with a letter.
<type> indicates the type the CreateObject function creates, as the assembly-qualified name of the type to create.
<text> any string of characters

The Template / x-script is composed by lines of instructions. Instructions are separated by "\r\n" (new line characters) or ";" character. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails. The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object.

An x-script instruction/line can be one of the following:

- **Dim variable**[, variable, ...] *declares the variables in the context. Multiple variables are separated by commas. The SetTemplateDef method can declare new variables to be available for the main context. (Sample: Dim h, h1, h2)*
- **variable =** [object.][property/method(arguments).]**property/method(arguments)**

*assigns the result of the **property/method** call to the **variable**. (Sample: h = Nodes.Add(`Node`))*

- [object.][property/method(arguments).]**property(arguments) = value** *assigns the **value** to the **property.** (Sample: Nodes.Add(`Node`).BackColor = RGB(255,0,0))*
- [object.][property/method(arguments).]**property/method(arguments)** *invokes the **property/method**. (Sample: Nodes.Add(`Node`))*
- **{**context **}** *delimits the object's context. The properties/fields or methods called between { and } are related to the last object returned by the property/method prior to { declaration. (Sample: Nodes{Add(`Child 1`);Add(`Child 2`)})*
- ***.** delimits the object than its property or method. (Sample: Nodes.Add(`Element`), or Nodes.Add(`Element`) and Nodes{Add(`Element`)} are equivalents)*

where

- ***variable** is the name of a variable declared with Dim command or previously defined using the SetTemplateDef method.*
- ***property** is the name of a property/field of the current object in the current context.*
- ***method** is the name of a method of the current object in the current context.*
- ***arguments** include constants and/or variables and/or property/method calls separated by comma character.*
- ***object** can be a variable of an Object type, **Me** or **CreateObject** call.*

The x-script uses constant expressions as follows:

- *boolean* expression with possible values as ***True*** or ***False**. The True value is equivalent with -1, while False with 0. (Sample: Visible = False)*
- *numeric* expression may starts with **0x** which indicates a hexa decimal representation, else it should starts with digit, or +/- followed by a digit, and . is the decimal separator. *Sample: 13 indicates the integer 13, or **12.45** indicates the double expression 12,45. (Sample: BackColor = 0xFF0000)*
- *date* expression is delimited by **#** character in the format **#mm/dd/yyyy hh:mm:ss#**. *Sample: #31/12/1971# indicates the December 31, 1971 (Sample: FirstVisibleDate = #1/1/2001#)*
- *string* expression is delimited by **"** or ` characters. If using the ` character, please make sure that it is different than ' which allows adding comments inline. *Sample: **"text"** or **`text`** indicates the string text, while the ' text , specifies the comment text. (Sample: Text = "caption")*

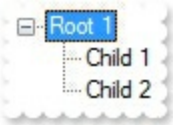
Also , the template or x-script code supports general functions as follows:

- **Me** *property indicates the original object, and it is defined as a predefined variable. (Sample: Me.Nodes.Add(`Root 1`))*
- **RGB(**R,G,B**)** *property retrieves an RGB value, where the R, G, B are byte values that*

indicates the Red Green Blue bytes for the color being specified. (Sample: Nodes.Add(`Root 1`).BackColor = RGB(255,0,0))

- **LoadPicture(**file**)** *property loads a picture from a file and returns a Picture object required by the picture properties. (Sample: BackgroundImage = LoadPicture(`C:\exontrol\images\auction.gif`)*
- **CreateObject(***assemblyQualifiedName***)** *property creates an instance of the specified type using that type's default constructor. The assemblyQualifiedName indicates the assembly-qualified name of the type to get. See AssemblyQualifiedName. If the type is in the currently executing assembly or in Mscorlib.dll, it is sufficient to supply the type name qualified by its namespace. (Sample: "CreateObject(`System.Windows.Forms.TabPage, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089`){Text = `Page`;UseVisualStyleBackColor = True}")*

The following samples shows how you can use the Template and TemplateResult properties:



VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`)"
      With .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      End With
   End With
End With
```

VB6

```
With NETHost1
```

```
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        .Template = "Nodes.Add(`Root 1`)"
        With .TemplateResult
            .Template = "Nodes.Add(`Child 1`)"
            .Template = "Nodes.Add(`Child 2`)"
            .Template = "Expand()"
        End With
    End With
End With
```

VB.NET

```
With Exnethost1
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        .Template = "Nodes.Add(`Root 1`)"
        With .TemplateResult
            .Template = "Nodes.Add(`Child 1`)"
            .Template = "Nodes.Add(`Child 2`)"
            .Template = "Expand()"
        End With
    End With
End With
```

VB.NET for /COM

```
With AxNETHost1
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
```

```
    With .Host
        .Template = "Nodes.Add(`Root 1`)"
        With .TemplateResult
            .Template = "Nodes.Add(`Child 1`)"
            .Template = "Nodes.Add(`Child 2`)"
            .Template = "Expand()"
        End With
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
>GetHost();
    var_NETHostObject->PutTemplate(L"Nodes.Add(`Root 1`)");
    exontrol_NETHost::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->GetTemplateResult();
        var_NETHostObject1->PutTemplate(L"Nodes.Add(`Child 1`)");
        var_NETHostObject1->PutTemplate(L"Nodes.Add(`Child 2`)");
        var_NETHostObject1->PutTemplate(L"Expand()");
```

C++ Builder

```
NETHost1->AssemblyLocation =
```

```
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   var_NETHostObject->Template = L"Nodes.Add(`Root 1`)";
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETHostObject1 =
var_NETHostObject->TemplateResult;
      var_NETHostObject1->Template = L"Nodes.Add(`Child 1`)";
      var_NETHostObject1->Template = L"Nodes.Add(`Child 2`)";
      var_NETHostObject1->Template = L"Expand()";
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   var_NETHostObject.Template = "Nodes.Add(`Root 1`)";
   exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject.TemplateResult;
      var_NETHostObject1.Template = "Nodes.Add(`Child 1`)";
      var_NETHostObject1.Template = "Nodes.Add(`Child 2`)";
      var_NETHostObject1.Template = "Expand()";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193
```

```
    NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
    var var_NETHostObject = NETHost1.Host;
        var_NETHostObject.Template = "Nodes.Add(`Root 1`)";
        var var_NETHostObject1 = var_NETHostObject.TemplateResult;
            var_NETHostObject1.Template = "Nodes.Add(`Child 1`)";
            var_NETHostObject1.Template = "Nodes.Add(`Child 2`)";
            var_NETHostObject1.Template = "Expand()";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
    With NETHost1
        .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

        .AssemblyName = "System.Windows.Forms.TreeView"
        With .Host
            .Template = "Nodes.Add(`Root 1`)"
            With .TemplateResult
                .Template = "Nodes.Add(`Child 1`)"
                .Template = "Nodes.Add(`Child 2`)"
                .Template = "Expand()"
            End With
        End With
    End With
End Function
</SCRIPT>
```

```
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
   var_NETHostObject.Template = "Nodes.Add(`Root 1`)";
   exontrol_NETHost.NETHostObject var_NETHostObject1 =
var_NETHostObject.TemplateResult;
      var_NETHostObject1.Template = "Nodes.Add(`Child 1`)";
      var_NETHostObject1.Template = "Nodes.Add(`Child 2`)";
      var_NETHostObject1.Template = "Expand()";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETHostObject,com_NETHostObject1;
   anytype var_NETHostObject,var_NETHostObject1;
   ;

   super();



exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
      com_NETHostObject.Template("Nodes.Add(`Root 1`)");
      var_NETHostObject1 = com_NETHostObject.TemplateResult();
com_NETHostObject1 = var_NETHostObject1;
         com_NETHostObject1.Template("Nodes.Add(`Child 1`)");
```

```
        com_NETHostObject1.Template("Nodes.Add(`Child 2`)");
        com_NETHostObject1.Template("Expand()");
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
  AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    Template := 'Nodes.Add(`Root 1`)';
    with TemplateResult do
    begin
      Template := 'Nodes.Add(`Child 1`)';
      Template := 'Nodes.Add(`Child 2`)';
      Template := 'Expand()';
    end;
  end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
  AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    Template := 'Nodes.Add(`Root 1`)';
    with TemplateResult do
    begin
      Template := 'Nodes.Add(`Child 1`)';
```

```
        Template := 'Nodes.Add(`Child 2`)';
        Template := 'Expand()';
      end;
    end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      .Template = "Nodes.Add(`Root 1`)"
      with .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      endwith
   endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETHostObject,var_NETHostObject1

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`)"
   var_NETHostObject1 = var_NETHostObject.TemplateResult
      var_NETHostObject1.Template = "Nodes.Add(`Child 1`)"
      var_NETHostObject1.Template = "Nodes.Add(`Child 2`)"
      var_NETHostObject1.Template = "Expand()"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P
Dim var_NETHostObject1 as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`)"
   var_NETHostObject1 = var_NETHostObject.TemplateResult
      var_NETHostObject1.Template = "Nodes.Add(`Child 1`)"
      var_NETHostObject1.Template = "Nodes.Add(`Child 2`)"
      var_NETHostObject1.Template = "Expand()"
```

Visual Objects

```
local var_NETHostObject as INETHostObject
local var_NETHostObject1 as INETObjectTemplate

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject:Template := "Nodes.Add(`Root 1`)"
   var_NETHostObject1 := var_NETHostObject:TemplateResult
      var_NETHostObject1:Template := "Nodes.Add(`Child 1`)"
      var_NETHostObject1:Template := "Nodes.Add(`Child 2`)"
      var_NETHostObject1:Template := "Expand()"
```

PowerBuilder

```
OleObject oNETHost,var_NETHostObject,var_NETHostObject1

oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`)"
   var_NETHostObject1 = var_NETHostObject.TemplateResult
      var_NETHostObject1.Template = "Nodes.Add(`Child 1`)"
      var_NETHostObject1.Template = "Nodes.Add(`Child 2`)"
      var_NETHostObject1.Template = "Expand()"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Set ComTemplate of hoNETHostObject to "Nodes.Add(`Root 1`)"
      Variant voNETHostObject1
      Get ComTemplateResult of hoNETHostObject to voNETHostObject1
      Handle hoNETHostObject1
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
      Set pvComObject of hoNETHostObject1 to voNETHostObject1
         Set ComTemplate of hoNETHostObject1 to "Nodes.Add(`Child 1`)"
         Set ComTemplate of hoNETHostObject1 to "Nodes.Add(`Child 2`)"
```

```
        Set ComTemplate of hoNETHostObject1 to "Expand()"
      Send Destroy to hoNETHostObject1
   Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHostObject1
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETHostObject := oNETHost:Host()
         oNETHostObject:Template := "Nodes.Add(`Root 1`)"
         oNETHostObject1 := oNETHostObject:TemplateResult()
            oNETHostObject1:Template := "Nodes.Add(`Child 1`)"
            oNETHostObject1:Template := "Nodes.Add(`Child 2`)"
            oNETHostObject1:Template := "Expand()"
```

```
   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

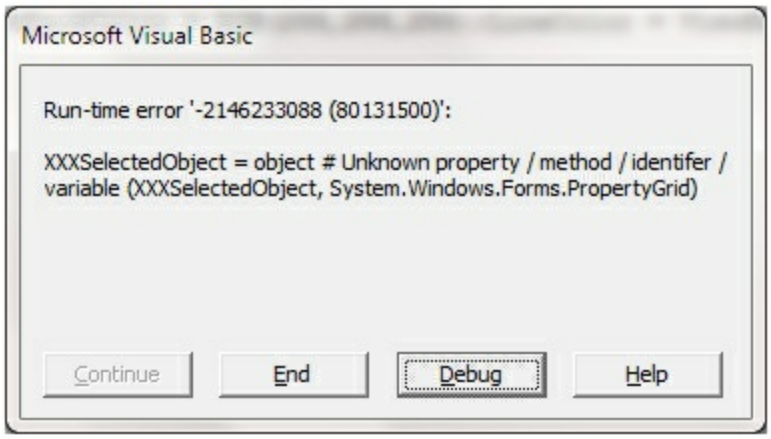
property NETHostObject.TemplateError as Long

Indicates the error code of the last Template call.

Type	Description
Long	A long expression that describes the error/exception in the Item/Template call.

By default, the TemplateError property returns 0. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

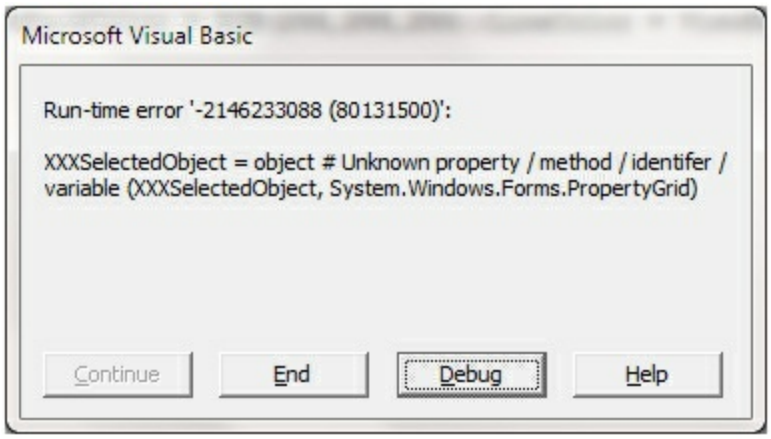
property NETHostObject.TemplateException as String

Indicates the detailed information about the exception that occurs.

Type	Description
String	A String expression that describes the error/exception in the Item/Template call.

By default, the TemplateException property is empty. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

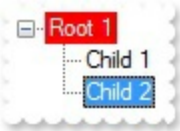
property NETHostObject.TemplateResult as NETObjectTemplate

Indicates the result of the last Template call.

Type	Description
NETObjectTemplate	A NETObjectTemplate object that holds the result of the last Template call.

The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object. Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

The following samples shows how you can use the Template and TemplateResult properties:



How can I use the TemplateResult property?

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB6

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB.NET

```
With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB.NET for /COM

```
With AxNETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
```

```
End With
```

C++

```
/*
   Copy and paste the following directives to your header file as
   it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

   #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETHostObjectPtr var_NETHostObject = spNETHost1-
>GetHost();
   var_NETHostObject->PutTemplate(L"Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }");
   var_NETHostObject->GetTemplateResult()->PutTemplate(L"Nodes{ Add(`Child
1`); Add(`Child 2`) }; Expand() }");
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETHostObjectPtr var_NETHostObject = NETHost1->Host;
   var_NETHostObject->Template = L"Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETHostObject->TemplateResult->Template = L"Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = exnethost1.Host;
   var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETHostObject = NETHost1.Host;
      var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
      var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
      With .Host
         .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
         .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand()
}"
      End With
   End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETHostObject var_NETHostObject = axNETHost1.Host;
   var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETHostObject,com_NETHostObject1;
   anytype var_NETHostObject,var_NETHostObject1;
   ;

   super();



exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETHostObject = exnethost1.Host(); com_NETHostObject =
var_NETHostObject;
      com_NETHostObject.Template("Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }");
      var_NETHostObject1 =
COM::createFromObject(com_NETHostObject.TemplateResult());
com_NETHostObject1 = var_NETHostObject1;
      com_NETHostObject1.Template("Nodes{ Add(`Child 1`); Add(`Child 2`) };
Expand() }");
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }';
      TemplateResult.Template := 'Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }';
   end;
```

```
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }';
      TemplateResult.Template := 'Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }';
   end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETHostObject

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETHostObject as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
```

Visual Objects

```
local var_NETHostObject as INETHostObject

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETHostObject := oDCOCX_Exontrol1:Host
   var_NETHostObject:Template := "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
```

```
var_NETHostObject:TemplateResult:Template := "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
```

PowerBuilder

```
OleObject oNETHost,var_NETHostObject

oNETHost = ole_1.Object
oNETHost.AssemblyLocation = "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETHostObject = oNETHost.Host
   var_NETHostObject.Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETHostObject.TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETHostObject
   Get ComHost to voNETHostObject
   Handle hoNETHostObject
   Get Create (RefClass(cComNETHostObject)) to hoNETHostObject
   Set pvComObject of hoNETHostObject to voNETHostObject
      Set ComTemplate of hoNETHostObject to "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor = RGB(255,255,255) }"
      Variant voNETHostObject1
      Get ComTemplateResult of hoNETHostObject to voNETHostObject1
      Handle hoNETHostObject1
```

```
      Get Create (RefClass(cComNETHostObject)) to hoNETHostObject1
      Set pvComObject of hoNETHostObject1 to voNETHostObject1
         Set ComTemplate of hoNETHostObject1 to "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
      Send Destroy to hoNETHostObject1
   Send Destroy to hoNETHostObject
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETHostObject
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETHostObject := oNETHost:Host()
         oNETHostObject:Template := "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
         oNETHostObject:TemplateResult():Template := "Nodes{ Add(`Child 1`);
```

```
Add(`Child 2`) }; Expand() }"

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

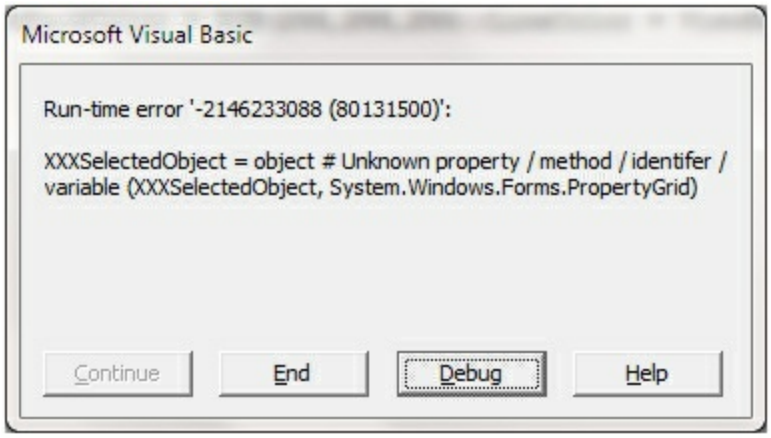
property NETHostObject.TemplateThrowError as Boolean

Specifies whether the execution of the template stops once an error occurs.

Type	Description
Boolean	A Boolean expression that specifies whether the NETHost control fires an error/exception when an error occurs in the Item/Template call.

By default, the TemplateThrowError property is True. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

property NETHostObject.Type as String

Indicates the type of the object's value.

Type	Description
String	A string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. For instance: "System.Windows.Forms.TreeView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property. The Value property holds the original object.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.Value as Variant

Specifies the value of the object.

Type	Description
Variant	A VARIANT expression that specifies the original object (.NET Framework object) hold by the current NETHostObject object.

The Value property holds the original object. Use the SetValue property to change the object being hosted by the current NETHostObject object. The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETHostObject.VtType as NETHostVarEnum

Indicates the type/vartype of the object's value.

Type	Description
NETHostVarEnum	A NETHostVarEnum expression that specifies the VARIANT-type of the Value.

The VtType property indicates the VARIANT type of the object that the current NETHostObject object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETHostObject holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETHostObject properties like: Item, SetTemplateDef or Template property. The Value property holds the original object.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

NETObjectTemplate object

The NETObjectTemplate object holds a .NET Framework object. The NETObjectTemplate type supports the following properties and method:

Name	Description
AsBoolean	Returns a boolean value that represents the current object's value.
AsDate	Returns a date value that represents the current object's value.
AsDouble	Returns a numeric value that represents the current object's value.
AsInt	Returns an integer value that represents the current object's value.
AsString	Returns a string that represents the current object's value.
Item	Executes the template and returns the result.
SetTemplateDef	Defines inside variables for the next Template/ExecuteTemplate call.
SetValue	Specifies the value of the object.
Template	Executes the x-script code.
TemplateError	Indicates the error code of the last Template call.
TemplateException	Indicates the detailed information about the exception that occurs.
TemplateResult	Indicates the result of the last Template call.
TemplateThrowError	Specifies whether the execution of the template stops once an error occurs.
Type	Indicates the type of the object's value.
Value	Specifies the value of the object.
VtType	Indicates the type/vartype of the object's value.

property NETObjectTemplate.AsBoolean as Boolean

Returns a boolean value that represents the current object's value.

Type	Description
Boolean	A Boolean expression that specifies the Value converted as boolean.

The AsBoolean property converts the Value to a Boolean expression. If the conversion is not possible, the AsBoolean property returns False. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.AsDate as Date

Returns a date value that represents the current object's value.

Type	Description
Date	A Date/Double expression that specifies the Value converted as date-time.

The AsDate property converts the Value to a DATE-TIME expression. If the conversion is not possible, the AsDate property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.AsDouble as Double

Returns a numeric value that represents the current object's value.

Type	Description
Double	A Double expression that specifies the Value converted as double.

The AsDouble property converts the Value to a double expression. If the conversion is not possible, the AsDouble property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.AsInt as Long

Returns an integer value that represents the current object's value.

Type	Description
Long	A Long expression that specifies the Value converted as long (32-bit integer).

The AsInt property converts the Value to a long expression. If the conversion is not possible, the AsInt property returns 0. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.AsString as String

Returns a string that represents the current object's value.

Type	Description
String	A String expression that specifies the Value converted as string.

The AsString property converts the Value to a string expression. If the conversion is not possible, the AsString property returns "" (empty). The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.Item (Template as String) as NETObjectTemplate

Executes the template and returns the result.

Type	Description
Template as String	A String expression that specifies the x-script/template code to be executed.
NETObjectTemplate	A NETObjectTemplate property that holds the result of the last instruction within the Template.

Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. Prior to Template/Item call, you can invoke the SetTemplateDef to define values from your code to Template's code (TemplateDef variables).

in VB/NET under the .NET Framework, you use a code like follows to add nodes to a TreeView control:

```
With TreeView1
  With .Nodes.Add("Root 1")
    .Nodes.Add("Child 1")
    With .Nodes.Add("Child 2")
      .Nodes.Add("Sub-Child 2.1")
      .Nodes.Add("Sub-Child 2.2")
      .Nodes.Add("Sub-Child 2.3")
      .Expand()
    End With
    .Nodes.Add("Child 3")
    .Expand()
  End With
  With .Nodes.Add("Root 2")
    .Nodes.Add("Child 1")
    .Nodes.Add("Child 2")
    .Nodes.Add("Child 3")
  End With
End With
```

while on VB using the NETHost control you should use a code like:

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         .Template = "Nodes.Add(`Child 1`)"
         With .Item("Nodes.Add(`Child 2`)")
            .Template = "Nodes.Add(`Sub-Child 2.1`)"
            .Template = "Nodes.Add(`Sub-Child 2.2`)"
            .Template = "Nodes.Add(`Sub-Child 2.3`)"
            .Template = "Expand()"
         End With
         .Template = "Nodes.Add(`Child 3`)"
         .Template = "Expand()"
      End With
      With .Item("Nodes.Add(`Root 2`)")
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Nodes.Add(`Child 3`)"
      End With
   End With
End With
```

The Template/ x-script code is a simple way of calling hosting control/object's properties, methods/ events using strings. Exontrol owns the x-script implementation in its easiest way and it does not require any VB engine to get executed. Our simple rule is using the component alone without any other dependency than the Windows system.

The **Template/x-script** syntax in BNF notation is defined like follows:

```
<x-script> := <lines>
<lines> := <line>[<eol> <lines>] | <block>
<block> := <call> [<eol>] { [<eol>] <lines> [<eol>] } [<eol>]
<eol> := ";" | "\r\n"
<line> := <dim> | <createobject> | <call> | <set> | <comment>
```

```
<dim> := "DIM" <variables>
<variables> := <variable> [, <variables>]
<variable> := "ME" | <identifier>
<createobject> := "CREATEOBJECT(`"<type>"`)"
<call> := <variable> | <property> | <variable>"."<property> | <createobject>"."
<property>
<property> := [<property>"."]<identifier>["("<parameters>")"]
<set> := <call> "=" <value>
<property> := <identifier> | <identifier>"("[<parameters>]")"
<parameters> := <value> [","<parameters>]
<value> := <boolean> | <number> | <color> | <date> | <string> | <createobject> |
<call>
<boolean> := "TRUE" | "FALSE"
<number> := "0X"<hexa> | ["-"]<integer>["."<integer>]
<digit10> := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<digit16> := <digit10> | A | B | C | D | E | F
<integer> := <digit10>[<integer>]
<hexa> := <digit16>[<hexa>]
<color> := "RGB("<integer>","<integer>","<integer>")"
<date> := "#"<integer>"/"<integer>"/"<integer>" "[<integer>":"<integer>":"
<integer>"]"#"
<string> := '"'<text>'"' | "`"<text>"`"
<comment> := "'"<text>
```

where:

<identifier> indicates an identifier of the variable, property or method, and should start with a letter.
<type> indicates the type the CreateObject function creates, as the assembly-qualified name of the type to create.
<text> any string of characters

The Template / x-script is composed by lines of instructions. Instructions are separated by "\r\n" (new line characters) or ";" character. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails. The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object.

An x-script instruction/line can be one of the following:

- **Dim variable**[, variable, ...] *declares the variables in the context. Multiple variables are separated by commas. The SetTemplateDef method can declare new variables to be available for the main context. (Sample: Dim h, h1, h2)*
- **variable =** [object.][property/method(arguments).]**property/method(arguments)** *assigns the result of the **property/method** call to the **variable**. (Sample: h = Nodes.Add(`Node`))*
- [object.][property/method(arguments).]**property(arguments) = value** *assigns the **value** to the **property.** (Sample: Nodes.Add(`Node`).BackColor = RGB(255,0,0))*
- [object.][property/method(arguments).]**property/method(arguments)** *invokes the **property/method**. (Sample: Nodes.Add(`Node`))*
- **{**context **}** *delimits the object's context. The properties/fields or methods called between { and } are related to the last object returned by the property/method prior to { declaration. (Sample: Nodes{Add(`Child 1`);Add(`Child 2`)})*
- ***.*** *delimits the object than its property or method. (Sample: Nodes.Add(`Element`), or Nodes.Add(`Element`) and Nodes{Add(`Element`)} are equivalents)*

where

- ***variable*** *is the name of a variable declared with Dim command or previously defined using the SetTemplateDef method.*
- ***property*** *is the name of a property/field of the current object in the current context.*
- ***method*** *is the name of a method of the current object in the current context.*
- ***arguments*** *include constants and/or variables and/or property/method calls separated by comma character.*
- ***object*** *can be a variable of an Object type, **Me** or **CreateObject** call.*

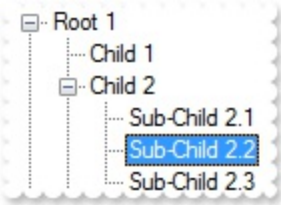
The x-script uses constant expressions as follows:

- *boolean* expression with possible values as ***True*** or ***False****. The True value is equivalent with -1, while False with 0. (Sample: Visible = False)*
- *numeric* expression may starts with **0x** which indicates a hexa decimal representation, else it should starts with digit, or +/- followed by a digit, and . is the decimal separator. *Sample: 13 indicates the integer 13, or **12.45** indicates the double expression 12,45. (Sample: BackColor = 0xFF0000)*
- *date* expression is delimited by **#** character in the format **#mm/dd/yyyy hh:mm:ss#**. *Sample: #31/12/1971# indicates the December 31, 1971 (Sample: FirstVisibleDate = #1/1/2001#)*
- *string* expression is delimited by **"** or ` characters. If using the ` character, please make sure that it is different than ' which allows adding comments inline. *Sample: **"text"** or **`text`** indicates the string text, while the ' text , specifies the comment text. (Sample: Text = "caption")*

Also , the template or x-script code supports general functions as follows:

- **Me** *property indicates the original object, and it is defined as a predefined variable. (Sample: Me.Nodes.Add(`Root 1`))*
- **RGB(**R,G,B**)** *property retrieves an RGB value, where the R, G, B are byte values that indicates the Red Green Blue bytes for the color being specified. (Sample: Nodes.Add(`Root 1`).BackColor = RGB(255,0,0))*
- **LoadPicture(**file**)** *property loads a picture from a file and returns a Picture object required by the picture properties. (Sample: BackgroundImage = LoadPicture(`C:\exontrol\images\auction.gif`)*
- **CreateObject(***assemblyQualifiedName***)** *property creates an instance of the specified type using that type's default constructor. The assemblyQualifiedName indicates the assembly-qualified name of the type to get. See AssemblyQualifiedName. If the type is in the currently executing assembly or in Mscorlib.dll, it is sufficient to supply the type name qualified by its namespace. (Sample: "CreateObject(`System.Windows.Forms.TabPage, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089`){Text = `Page`;UseVisualStyleBackColor = True}")*

The following samples shows how you can use the Item property.



VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         Set var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
         With .Item("Nodes.Add(`Child 2`)")
            Set var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            Set var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            Set var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            Set var_Object = .Item("Expand()")
```

```
        End With
        Set var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
        Set var_Object1 = .Item("Expand()")
    End With
    With .Item("Nodes.Add(`Root 2`)")
        Set var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
        Set var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
        Set var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
    End With
  End With
End With
```

VB6

```
With NETHost1
  .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  .AssemblyName = "System.Windows.Forms.TreeView"
  With .Host
    With .Item("Nodes.Add(`Root 1`)")
      Set var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
      With .Item("Nodes.Add(`Child 2`)")
        Set var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
        Set var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
        Set var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
        Set var_Object = .Item("Expand()")
      End With
      Set var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
      Set var_Object1 = .Item("Expand()")
    End With
    With .Item("Nodes.Add(`Root 2`)")
      Set var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
      Set var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
      Set var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
    End With
  End With
```

```
End With
```

VB.NET

```
Dim
var_NETObjectTemplate,var_NETObjectTemplate1,var_NETObjectTemplate2,var_NETObj

With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
         With .Item("Nodes.Add(`Child 2`)")
            var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            var_Object = .Item("Expand()")
         End With
         var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
         var_Object1 = .Item("Expand()")
      End With
      With .Item("Nodes.Add(`Root 2`)")
         var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
         var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
         var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
      End With
   End With
End With
```

VB.NET for /COM

```
Dim
var_NETObjectTemplate,var_NETObjectTemplate1,var_NETObjectTemplate2,var_NETObj

With AxNETHost1
```

```
    .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

    .AssemblyName = "System.Windows.Forms.TreeView"
    With .Host
        With .Item("Nodes.Add(`Root 1`)")
            var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
            With .Item("Nodes.Add(`Child 2`)")
                var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
                var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
                var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
                var_Object = .Item("Expand()")
            End With
            var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
            var_Object1 = .Item("Expand()")
        End With
        With .Item("Nodes.Add(`Root 2`)")
            var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
            var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
            var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
        End With
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form
```

```
spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate = spNETHost1-
>GetHost();
   exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate1 =
var_NETObjectTemplate->GetItem(L"Nodes.Add(`Root 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate =
var_NETObjectTemplate1->GetItem(L"Nodes.Add(`Child 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate2 =
var_NETObjectTemplate1->GetItem(L"Nodes.Add(`Child 2`)");
         var_NETObjectTemplate1 = var_NETObjectTemplate2-
>GetItem(L"Nodes.Add(`Sub-Child 2.1`)");
         var_NETObjectTemplate2 = var_NETObjectTemplate2-
>GetItem(L"Nodes.Add(`Sub-Child 2.2`)");
         exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate3 =
var_NETObjectTemplate2->GetItem(L"Nodes.Add(`Sub-Child 2.3`)");
         ObjectPtr var_Object = ((ObjectPtr)(var_NETObjectTemplate2-
>GetItem(L"Expand()")));
      exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate4 =
var_NETObjectTemplate1->GetItem(L"Nodes.Add(`Child 3`)");
      ObjectPtr var_Object1 = ((ObjectPtr)(var_NETObjectTemplate1-
>GetItem(L"Expand()")));
   exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate5 =
var_NETObjectTemplate->GetItem(L"Nodes.Add(`Root 2`)");
      var_NETObjectTemplate5 = var_NETObjectTemplate5-
>GetItem(L"Nodes.Add(`Child 1`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate6 =
var_NETObjectTemplate5->GetItem(L"Nodes.Add(`Child 2`)");
      exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate7 =
var_NETObjectTemplate5->GetItem(L"Nodes.Add(`Child 3`)");
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619
```

```
NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate = NETHost1-
>Host;
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate1 =
var_NETObjectTemplate->get_Item(L"Nodes.Add(`Root 1`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate =
var_NETObjectTemplate1->get_Item(L"Nodes.Add(`Child 1`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate2 =
var_NETObjectTemplate1->get_Item(L"Nodes.Add(`Child 2`)");
         var_NETObjectTemplate1 = var_NETObjectTemplate2-
>get_Item(L"Nodes.Add(`Sub-Child 2.1`)");
         var_NETObjectTemplate2 = var_NETObjectTemplate2-
>get_Item(L"Nodes.Add(`Sub-Child 2.2`)");
         Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate3 =
var_NETObjectTemplate2->get_Item(L"Nodes.Add(`Sub-Child 2.3`)");
         _tlb::ObjectPtr var_Object = var_NETObjectTemplate2->get_Item(L"Expand()");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate4 =
var_NETObjectTemplate1->get_Item(L"Nodes.Add(`Child 3`)");
      _tlb::ObjectPtr var_Object1 = var_NETObjectTemplate1->get_Item(L"Expand()");
   Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate5 =
var_NETObjectTemplate->get_Item(L"Nodes.Add(`Root 2`)");
      var_NETObjectTemplate5 = var_NETObjectTemplate5-
>get_Item(L"Nodes.Add(`Child 1`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate6 =
var_NETObjectTemplate5->get_Item(L"Nodes.Add(`Child 2`)");
      Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate7 =
var_NETObjectTemplate5->get_Item(L"Nodes.Add(`Child 3`)");
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = exnethost1.Host;
   exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate1 =
```

```
var_NETObjectTemplate["Nodes.Add(`Root 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate =
var_NETObjectTemplate1["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate2 =
var_NETObjectTemplate1["Nodes.Add(`Child 2`)"];
         var_NETObjectTemplate1 = var_NETObjectTemplate2["Nodes.Add(`Sub-Child
2.1`)"];
         var_NETObjectTemplate2 = var_NETObjectTemplate2["Nodes.Add(`Sub-Child
2.2`)"];
         exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate3 =
var_NETObjectTemplate2["Nodes.Add(`Sub-Child 2.3`)"];
         Object var_Object = (var_NETObjectTemplate2["Expand()"] as Object);
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate4 =
var_NETObjectTemplate1["Nodes.Add(`Child 3`)"];
      Object var_Object1 = (var_NETObjectTemplate1["Expand()"] as Object);
   exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate5 =
var_NETObjectTemplate["Nodes.Add(`Root 2`)"];
      var_NETObjectTemplate5 = var_NETObjectTemplate5["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate6 =
var_NETObjectTemplate5["Nodes.Add(`Child 2`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate7 =
var_NETObjectTemplate5["Nodes.Add(`Child 3`)"];
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
```

```
var var_NETObjectTemplate = NETHost1.Host;
var var_NETObjectTemplate1 = var_NETObjectTemplate.Item("Nodes.Add(`Root 1`)");
var var_NETObjectTemplate = var_NETObjectTemplate1.Item("Nodes.Add(`Child 1`)");
var var_NETObjectTemplate2 = var_NETObjectTemplate1.Item("Nodes.Add(`Child 2`)");
var_NETObjectTemplate1 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child 2.1`)");
var_NETObjectTemplate2 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child 2.2`)");
var var_NETObjectTemplate3 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child 2.3`)");
var var_Object = var_NETObjectTemplate2.Item("Expand()");
var var_NETObjectTemplate4 = var_NETObjectTemplate1.Item("Nodes.Add(`Child 3`)");
var var_Object1 = var_NETObjectTemplate1.Item("Expand()");
var var_NETObjectTemplate5 = var_NETObjectTemplate.Item("Nodes.Add(`Root 2`)");
var_NETObjectTemplate5 = var_NETObjectTemplate5.Item("Nodes.Add(`Child 1`)");
var var_NETObjectTemplate6 = var_NETObjectTemplate5.Item("Nodes.Add(`Child 2`)");
var var_NETObjectTemplate7 = var_NETObjectTemplate5.Item("Nodes.Add(`Child 3`)");
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
```

```
Function Init()
   With NETHost1
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
      With .Host
         With .Item("Nodes.Add(`Root 1`)")
            Set var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
            With .Item("Nodes.Add(`Child 2`)")
               Set var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
               Set var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
               Set var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
               Set var_Object = .Item("Expand()")
            End With
            Set var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
            Set var_Object1 = .Item("Expand()")
         End With
         With .Item("Nodes.Add(`Root 2`)")
            Set var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
            Set var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
            Set var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
         End With
      End With
   End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = axNETHost1.Host;
```

```
   exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate1 =
var_NETObjectTemplate["Nodes.Add(`Root 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate =
var_NETObjectTemplate1["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate2 =
var_NETObjectTemplate1["Nodes.Add(`Child 2`)"];
         var_NETObjectTemplate1 = var_NETObjectTemplate2["Nodes.Add(`Sub-Child
2.1`)"];
         var_NETObjectTemplate2 = var_NETObjectTemplate2["Nodes.Add(`Sub-Child
2.2`)"];
         exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate3 =
var_NETObjectTemplate2["Nodes.Add(`Sub-Child 2.3`)"];
         Object var_Object = (var_NETObjectTemplate2["Expand()"] as Object);
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate4 =
var_NETObjectTemplate1["Nodes.Add(`Child 3`)"];
      Object var_Object1 = (var_NETObjectTemplate1["Expand()"] as Object);
   exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate5 =
var_NETObjectTemplate["Nodes.Add(`Root 2`)"];
      var_NETObjectTemplate5 = var_NETObjectTemplate5["Nodes.Add(`Child 1`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate6 =
var_NETObjectTemplate5["Nodes.Add(`Child 2`)"];
      exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate7 =
var_NETObjectTemplate5["Nodes.Add(`Child 3`)"];
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM
com_NETObjectTemplate,com_NETObjectTemplate1,com_NETObjectTemplate2,com_NE

   anytype
var_NETObjectTemplate,var_NETObjectTemplate1,var_NETObjectTemplate2,var_NETObj

   ;
```

```
    super();


exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

    exnethost1.AssemblyName("System.Windows.Forms.TreeView");
    var_NETObjectTemplate = exnethost1.Host(); com_NETObjectTemplate =
var_NETObjectTemplate;
        var_NETObjectTemplate1 = com_NETObjectTemplate.Item("Nodes.Add(`Root
1`)"); com_NETObjectTemplate1 = var_NETObjectTemplate1;
            var_NETObjectTemplate =
COM::createFromObject(com_NETObjectTemplate1.Item("Nodes.Add(`Child 1`)"));
com_NETObjectTemplate = var_NETObjectTemplate;
            var_NETObjectTemplate2 = com_NETObjectTemplate1.Item("Nodes.Add(`Child
2`)"); com_NETObjectTemplate2 = var_NETObjectTemplate2;
                var_NETObjectTemplate1 =
COM::createFromObject(com_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child
2.1`)"));
                var_NETObjectTemplate2 =
COM::createFromObject(com_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child
2.2`)"));
                var_NETObjectTemplate3 =
COM::createFromObject(com_NETObjectTemplate2.Item("Nodes.Add(`Sub-Child
2.3`)")); com_NETObjectTemplate3 = var_NETObjectTemplate3;
                var_Object =
COM::createFromObject(com_NETObjectTemplate2.Item("Expand()")); com_Object =
var_Object;
            var_NETObjectTemplate4 =
COM::createFromObject(com_NETObjectTemplate1.Item("Nodes.Add(`Child 3`)"));
com_NETObjectTemplate4 = var_NETObjectTemplate4;
            var_Object1 =
COM::createFromObject(com_NETObjectTemplate1.Item("Expand()")); com_Object1 =
var_Object1;
        var_NETObjectTemplate5 = com_NETObjectTemplate.Item("Nodes.Add(`Root
2`)"); com_NETObjectTemplate5 = var_NETObjectTemplate5;
            var_NETObjectTemplate5 =
COM::createFromObject(com_NETObjectTemplate5.Item("Nodes.Add(`Child 1`)"));
```

```
        var_NETObjectTemplate6 =
COM::createFromObject(com_NETObjectTemplate5.Item("Nodes.Add(`Child 2`)"));
com_NETObjectTemplate6 = var_NETObjectTemplate6;
        var_NETObjectTemplate7 =
COM::createFromObject(com_NETObjectTemplate5.Item("Nodes.Add(`Child 3`)"));
com_NETObjectTemplate7 = var_NETObjectTemplate7;
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
  AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    with Item['Nodes.Add(`Root 1`)'] do
    begin
      var_NETObjectTemplate := Item['Nodes.Add(`Child 1`)'];
      with Item['Nodes.Add(`Child 2`)'] do
      begin
        var_NETObjectTemplate1 := Item['Nodes.Add(`Sub-Child 2.1`)'];
        var_NETObjectTemplate2 := Item['Nodes.Add(`Sub-Child 2.2`)'];
        var_NETObjectTemplate3 := Item['Nodes.Add(`Sub-Child 2.3`)'];
        var_Object := (Item['Expand()'] as Object);
      end;
      var_NETObjectTemplate4 := Item['Nodes.Add(`Child 3`)'];
      var_Object1 := (Item['Expand()'] as Object);
    end;
    with Item['Nodes.Add(`Root 2`)'] do
    begin
      var_NETObjectTemplate5 := Item['Nodes.Add(`Child 1`)'];
      var_NETObjectTemplate6 := Item['Nodes.Add(`Child 2`)'];
      var_NETObjectTemplate7 := Item['Nodes.Add(`Child 3`)'];
    end;
```

```
  end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
  AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    with Item['Nodes.Add(`Root 1`)'] do
    begin
      var_NETObjectTemplate := Item['Nodes.Add(`Child 1`)'];
      with Item['Nodes.Add(`Child 2`)'] do
      begin
        var_NETObjectTemplate1 := Item['Nodes.Add(`Sub-Child 2.1`)'];
        var_NETObjectTemplate2 := Item['Nodes.Add(`Sub-Child 2.2`)'];
        var_NETObjectTemplate3 := Item['Nodes.Add(`Sub-Child 2.3`)'];
        var_Object := (IUnknown(Item['Expand()']) as _TLB.Object);
      end;
      var_NETObjectTemplate4 := Item['Nodes.Add(`Child 3`)'];
      var_Object1 := (IUnknown(Item['Expand()']) as _TLB.Object);
    end;
    with Item['Nodes.Add(`Root 2`)'] do
    begin
      var_NETObjectTemplate5 := Item['Nodes.Add(`Child 1`)'];
      var_NETObjectTemplate6 := Item['Nodes.Add(`Child 2`)'];
      var_NETObjectTemplate7 := Item['Nodes.Add(`Child 3`)'];
    end;
  end;
end
```

VFP

```
with thisform.NETHost1
```

```
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      with .Item("Nodes.Add(`Root 1`)")
         var_NETObjectTemplate = .Item("Nodes.Add(`Child 1`)")
         with .Item("Nodes.Add(`Child 2`)")
            var_NETObjectTemplate1 = .Item("Nodes.Add(`Sub-Child 2.1`)")
            var_NETObjectTemplate2 = .Item("Nodes.Add(`Sub-Child 2.2`)")
            var_NETObjectTemplate3 = .Item("Nodes.Add(`Sub-Child 2.3`)")
            var_Object = .Item("Expand()")
         endwith
         var_NETObjectTemplate4 = .Item("Nodes.Add(`Child 3`)")
         var_Object1 = .Item("Expand()")
      endwith
      with .Item("Nodes.Add(`Root 2`)")
         var_NETObjectTemplate5 = .Item("Nodes.Add(`Child 1`)")
         var_NETObjectTemplate6 = .Item("Nodes.Add(`Child 2`)")
         var_NETObjectTemplate7 = .Item("Nodes.Add(`Child 3`)")
      endwith
   endwith
endwith
```

dBASE Plus

```
local
oNETHost,var_NETObjectTemplate,var_NETObjectTemplate1,var_NETObjectTemplate2,v



oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate1 = var_NETObjectTemplate.Item("Nodes.Add(`Root 1`)")
```

```
    var_NETObjectTemplate = var_NETObjectTemplate1.Item("Nodes.Add(`Child 1`)")
    var_NETObjectTemplate2 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
2`)")
      var_NETObjectTemplate1 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.1`)")
      var_NETObjectTemplate2 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.2`)")
      var_NETObjectTemplate3 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.3`)")
      var_Object = var_NETObjectTemplate2.Item("Expand()")
    var_NETObjectTemplate4 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
3`)")
    var_Object1 = var_NETObjectTemplate1.Item("Expand()")
  var_NETObjectTemplate5 = var_NETObjectTemplate.Item("Nodes.Add(`Root 2`)")
    var_NETObjectTemplate5 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
1`)")
    var_NETObjectTemplate6 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
2`)")
    var_NETObjectTemplate7 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
3`)")
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETObjectTemplate as P
Dim var_NETObjectTemplate1 as P
Dim var_NETObjectTemplate2 as P
Dim var_NETObjectTemplate3 as P
Dim var_NETObjectTemplate4 as P
Dim var_NETObjectTemplate5 as P
Dim var_NETObjectTemplate6 as P
Dim var_NETObjectTemplate7 as P
Dim var_Object as P
Dim var_Object1 as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
```

```
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate1 = var_NETObjectTemplate.Item("Nodes.Add(`Root 1`)")
      var_NETObjectTemplate = var_NETObjectTemplate1.Item("Nodes.Add(`Child 1`)")
      var_NETObjectTemplate2 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
2`)")
         var_NETObjectTemplate1 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.1`)")
         var_NETObjectTemplate2 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.2`)")
         var_NETObjectTemplate3 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.3`)")
         var_Object = var_NETObjectTemplate2.Item("Expand()")
      var_NETObjectTemplate4 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
3`)")
      var_Object1 = var_NETObjectTemplate1.Item("Expand()")
   var_NETObjectTemplate5 = var_NETObjectTemplate.Item("Nodes.Add(`Root 2`)")
      var_NETObjectTemplate5 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
1`)")
      var_NETObjectTemplate6 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
2`)")
      var_NETObjectTemplate7 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
3`)")
```

Visual Objects

```
local var_NETObjectTemplate as INETObjectTemplate
local
var_NETObjectTemplate1,var_NETObjectTemplate2,var_NETObjectTemplate3,var_NETOb
 as INETObjectTemplate
local var_Object,var_Object1 as USUAL

oDCOCX_Exontrol1:AssemblyLocation :=
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETObjectTemplate := oDCOCX_Exontrol1:Host
   var_NETObjectTemplate1 := var_NETObjectTemplate:[Item,"Nodes.Add(`Root 1`)"]
      var_NETObjectTemplate := var_NETObjectTemplate1:[Item,"Nodes.Add(`Child
1`)"]
      var_NETObjectTemplate2 := var_NETObjectTemplate1:[Item,"Nodes.Add(`Child
2`)"]
         var_NETObjectTemplate1 := var_NETObjectTemplate2:[Item,"Nodes.Add(`Sub-
Child 2.1`)"]
         var_NETObjectTemplate2 := var_NETObjectTemplate2:[Item,"Nodes.Add(`Sub-
Child 2.2`)"]
         var_NETObjectTemplate3 := var_NETObjectTemplate2:[Item,"Nodes.Add(`Sub-
Child 2.3`)"]
         var_Object := var_NETObjectTemplate2:[Item,"Expand()"]
      var_NETObjectTemplate4 := var_NETObjectTemplate1:[Item,"Nodes.Add(`Child
3`)"]
      var_Object1 := var_NETObjectTemplate1:[Item,"Expand()"]
   var_NETObjectTemplate5 := var_NETObjectTemplate:[Item,"Nodes.Add(`Root 2`)"]
      var_NETObjectTemplate5 := var_NETObjectTemplate5:[Item,"Nodes.Add(`Child
1`)"]
      var_NETObjectTemplate6 := var_NETObjectTemplate5:[Item,"Nodes.Add(`Child
2`)"]
      var_NETObjectTemplate7 := var_NETObjectTemplate5:[Item,"Nodes.Add(`Child
3`)"]
```

PowerBuilder

```
OleObject
oNETHost,var_NETObjectTemplate,var_NETObjectTemplate1,var_NETObjectTemplate2,v



oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate1 = var_NETObjectTemplate.Item("Nodes.Add(`Root 1`)")
      var_NETObjectTemplate = var_NETObjectTemplate1.Item("Nodes.Add(`Child 1`)")
      var_NETObjectTemplate2 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
2`)")
         var_NETObjectTemplate1 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.1`)")
         var_NETObjectTemplate2 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.2`)")
         var_NETObjectTemplate3 = var_NETObjectTemplate2.Item("Nodes.Add(`Sub-
Child 2.3`)")
         var_Object = var_NETObjectTemplate2.Item("Expand()")
      var_NETObjectTemplate4 = var_NETObjectTemplate1.Item("Nodes.Add(`Child
3`)")
      var_Object1 = var_NETObjectTemplate1.Item("Expand()")
   var_NETObjectTemplate5 = var_NETObjectTemplate.Item("Nodes.Add(`Root 2`)")
      var_NETObjectTemplate5 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
1`)")
      var_NETObjectTemplate6 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
2`)")
      var_NETObjectTemplate7 = var_NETObjectTemplate5.Item("Nodes.Add(`Child
3`)")
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETObjectTemplate
   Get ComHost to voNETObjectTemplate
   Handle hoNETObjectTemplate
```

```
Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate
Set pvComObject of hoNETObjectTemplate to voNETObjectTemplate
    Variant voNETObjectTemplate1
    Get ComItem of hoNETObjectTemplate "Nodes.Add(`Root 1`)" to
voNETObjectTemplate1
    Handle hoNETObjectTemplate1
    Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate1
    Set pvComObject of hoNETObjectTemplate1 to voNETObjectTemplate1
        Variant var_NETObjectTemplate
        Get ComItem of hoNETObjectTemplate1 "Nodes.Add(`Child 1`)" to
var_NETObjectTemplate
        Variant voNETObjectTemplate2
        Get ComItem of hoNETObjectTemplate1 "Nodes.Add(`Child 2`)" to
voNETObjectTemplate2
        Handle hoNETObjectTemplate2
        Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate2
        Set pvComObject of hoNETObjectTemplate2 to voNETObjectTemplate2
            Variant var_NETObjectTemplate1
            Get ComItem of hoNETObjectTemplate2 "Nodes.Add(`Sub-Child 2.1`)" to
var_NETObjectTemplate1
            Variant var_NETObjectTemplate2
            Get ComItem of hoNETObjectTemplate2 "Nodes.Add(`Sub-Child 2.2`)" to
var_NETObjectTemplate2
            Variant var_NETObjectTemplate3
            Get ComItem of hoNETObjectTemplate2 "Nodes.Add(`Sub-Child 2.3`)" to
var_NETObjectTemplate3
            Variant var_Object
            Get ComItem of hoNETObjectTemplate2 "Expand()" to var_Object
        Send Destroy to hoNETObjectTemplate2
        Variant var_NETObjectTemplate4
        Get ComItem of hoNETObjectTemplate1 "Nodes.Add(`Child 3`)" to
var_NETObjectTemplate4
        Variant var_Object1
        Get ComItem of hoNETObjectTemplate1 "Expand()" to var_Object1
    Send Destroy to hoNETObjectTemplate1
    Variant voNETObjectTemplate3
    Get ComItem of hoNETObjectTemplate "Nodes.Add(`Root 2`)" to
```

```
voNETObjectTemplate3
      Handle hoNETObjectTemplate3
      Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate3
      Set pvComObject of hoNETObjectTemplate3 to voNETObjectTemplate3
         Variant var_NETObjectTemplate5
         Get ComItem of hoNETObjectTemplate3 "Nodes.Add(`Child 1`)" to
var_NETObjectTemplate5
         Variant var_NETObjectTemplate6
         Get ComItem of hoNETObjectTemplate3 "Nodes.Add(`Child 2`)" to
var_NETObjectTemplate6
         Variant var_NETObjectTemplate7
         Get ComItem of hoNETObjectTemplate3 "Nodes.Add(`Child 3`)" to
var_NETObjectTemplate7
      Send Destroy to hoNETObjectTemplate3
   Send Destroy to hoNETObjectTemplate
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETObjectTemplate
   LOCAL
oNETObjectTemplate1,oNETObjectTemplate2,oNETObjectTemplate3,var_NETObjectTem

   LOCAL oNETHost
   LOCAL var_Object,var_Object1

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}
```

```
   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETObjectTemplate := oNETHost:Host()
         oNETObjectTemplate1 := oNETObjectTemplate:Item("Nodes.Add(`Root 1`)")
            var_NETObjectTemplate := oNETObjectTemplate1:Item("Nodes.Add(`Child
1`)")
            oNETObjectTemplate2 := oNETObjectTemplate1:Item("Nodes.Add(`Child
2`)")
               var_NETObjectTemplate1 :=
oNETObjectTemplate2:Item("Nodes.Add(`Sub-Child 2.1`)")
               var_NETObjectTemplate2 :=
oNETObjectTemplate2:Item("Nodes.Add(`Sub-Child 2.2`)")
               var_NETObjectTemplate3 :=
oNETObjectTemplate2:Item("Nodes.Add(`Sub-Child 2.3`)")
               var_Object := oNETObjectTemplate2:Item("Expand()")
            var_NETObjectTemplate4 := oNETObjectTemplate1:Item("Nodes.Add(`Child
3`)")
            var_Object1 := oNETObjectTemplate1:Item("Expand()")
         oNETObjectTemplate3 := oNETObjectTemplate:Item("Nodes.Add(`Root 2`)")
            var_NETObjectTemplate5 := oNETObjectTemplate3:Item("Nodes.Add(`Child
1`)")
            var_NETObjectTemplate6 := oNETObjectTemplate3:Item("Nodes.Add(`Child
2`)")
            var_NETObjectTemplate7 := oNETObjectTemplate3:Item("Nodes.Add(`Child
3`)")

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
```

```
   ENDDO
RETURN
```

method NETObjectTemplate.SetTemplateDef (Value as Variant)

Defines inside variables for the next Template/ExecuteTemplate call.

Type	Description
Value as Variant	If calling the first time, A String expression that indicates the DIM command to define the variables that follows, or a VARIANT expression that defines the value of the variable in the order as they were defined.

The SetTemplateDef method was provided to let you use values/objects inside the next Template/Item call. For instance, let's say you have a date field in your form, and once the user fills it, you want a /NET Frameworks MonthCalendar object to select it. In order to do that you have to call a code like:

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

This sample defines the variable x to be 1/1/2001 for the Template call, so the SelectionStart will be set on 1/1/2001.

The call of SetTemplateDef method consists in:

- **First** call of SetTemplateDef method should be on a form of SetTemplateDef("Dim variable[,variable,...]"). This defines the name of the variable that follow to be defined.
- **Next** calls, must be exactly the same as with the number of variables you defined, which will define the variable one by one. For instance, if your first call was SetTemplateDef("Dim h1,h2,h3"), it means that the next-three calls of SetTemplateDef defines the variable h1, h2 and h3

Once you defined the variables, they will be available for the next calls of Template/Item properties.

The following samples shows how you can define the x variable to be set on the SelectionStart property of the MonthCalendar object:

VBA (MS Access, Excell...)

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

VB6

```
With NETHost1
   .BackgroundColor = 16777215
   .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   With .Host
      .SetTemplateDef "Dim x"
      .SetTemplateDef #1/1/2001#
      .Template = "MaxSelectionCount = 1;SelectionStart = x"
   End With
End With
```

VB.NET

```
With Exnethost1
   .BackgroundColor = 16777215

.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56

   With .Host
```

```
        .SetTemplateDef("Dim x")
        .SetTemplateDef(#1/1/2001#)
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    End With
End With
```

VB.NET for /COM

```
With AxNETHost1
    .BackgroundColor = 16777215

.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56

    With .Host
        .SetTemplateDef("Dim x")
        .SetTemplateDef(#1/1/2001#)
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    End With
End With
```

C++

```
/*
    Copy and paste the following directives to your header file as
    it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

    #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1->PutBackgroundColor(16777215);
spNETHost1-
>Create(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77

exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate = spNETHost1-
>GetHost();
    var_NETObjectTemplate->SetTemplateDef("Dim x");
```

```
var_NETObjectTemplate->SetTemplateDef(COleDateTime(2001,1,1,0,00,00).operator DATE());
var_NETObjectTemplate->PutTemplate(L"MaxSelectionCount = 1;SelectionStart = x");
```

C++ Builder

```
NETHost1->BackgroundColor = 16777215;
NETHost1->Create(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77

Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate = NETHost1->Host;
var_NETObjectTemplate->SetTemplateDef(TVariant("Dim x"));
var_NETObjectTemplate->SetTemplateDef(TVariant(TDateTime(2001,1,1).operator double()));
var_NETObjectTemplate->Template = L"MaxSelectionCount = 1;SelectionStart = x";
```

C#

```
exnethost1.BackgroundColor = 16777215;
exnethost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0

exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = exnethost1.Host;
var_NETObjectTemplate.SetTemplateDef("Dim x");

var_NETObjectTemplate.SetTemplateDef(Convert.ToDateTime("1/1/2001",System.Globa
US")));
var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>
```

```
<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.BackgroundColor = 16777215;

NETHost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.(

   var var_NETObjectTemplate = NETHost1.Host;
      var_NETObjectTemplate.SetTemplateDef("Dim x");
      var_NETObjectTemplate.SetTemplateDef("1/1/2001");
      var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .BackgroundColor = 16777215
      .Create
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      With .Host
         .SetTemplateDef "Dim x"
         .SetTemplateDef #1/1/2001#
         .Template = "MaxSelectionCount = 1;SelectionStart = x"
      End With
   End With
End Function
```

```
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.BackgroundColor = 16777215;
axNETHost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.

exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = axNETHost1.Host;
   var_NETObjectTemplate.SetTemplateDef("Dim x");

var_NETObjectTemplate.SetTemplateDef(Convert.ToDateTime("1/1/2001",System.Globa
US")));
   var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETObjectTemplate;
   anytype var_NETObjectTemplate;
   ;

   super();

   exnethost1.BackgroundColor(16777215);

exnethost1.Create("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0

   var_NETObjectTemplate = exnethost1.Host(); com_NETObjectTemplate =
var_NETObjectTemplate;
      com_NETObjectTemplate.SetTemplateDef("Dim x");

com_NETObjectTemplate.SetTemplateDef(COMVariant::createFromDate(str2Date("1/1/2

      com_NETObjectTemplate.Template("MaxSelectionCount = 1;SelectionStart = x");
```

```
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   BackgroundColor := 16777215;

Create('C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561

   with Host do
   begin
      SetTemplateDef('Dim x');
      SetTemplateDef('1/1/2001');
      Template := 'MaxSelectionCount = 1;SelectionStart = x';
   end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
   BackgroundColor := 16777215;

Create('C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561

   with Host do
   begin
      SetTemplateDef('Dim x');
      SetTemplateDef('1/1/2001');
      Template := 'MaxSelectionCount = 1;SelectionStart = x';
   end;
end
```

VFP

```
with thisform.NETHost1
   .BackgroundColor = 16777215
```

```
.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c56

    with .Host
        .SetTemplateDef("Dim x")
        .SetTemplateDef({^2001-1-1})
        .Template = "MaxSelectionCount = 1;SelectionStart = x"
    endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETObjectTemplate

oNETHost = form.Activex1.nativeObject
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETObjectTemplate = oNETHost.Host
    var_NETObjectTemplate.SetTemplateDef("Dim x")
    var_NETObjectTemplate.SetTemplateDef("01/01/2001")
    var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETObjectTemplate as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETObjectTemplate = oNETHost.Host
    var_NETObjectTemplate.SetTemplateDef("Dim x")
    var_NETObjectTemplate.SetTemplateDef({01/01/2001})
    var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

Visual Objects

```
local var_NETObjectTemplate as INETObjectTemplate

oDCOCX_Exontrol1:BackgroundColor := 16777215
oDCOCX_Exontrol1:Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms

var_NETObjectTemplate := oDCOCX_Exontrol1:Host
   var_NETObjectTemplate:SetTemplateDef("Dim x")
   var_NETObjectTemplate:SetTemplateDef(SToD("20010101"))
   var_NETObjectTemplate:Template := "MaxSelectionCount = 1;SelectionStart = x"
```

PowerBuilder

```
OleObject oNETHost,var_NETObjectTemplate

oNETHost = ole_1.Object
oNETHost.BackgroundColor = 16777215
oNETHost.Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.SetTemplateDef("Dim x")
   var_NETObjectTemplate.SetTemplateDef(2001-01-01)
   var_NETObjectTemplate.Template = "MaxSelectionCount = 1;SelectionStart = x"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComBackgroundColor to 16777215
   Get ComCreate
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
 "System.Windows.Forms.MonthCalendar" to Nothing
   Variant voNETObjectTemplate
   Get ComHost to voNETObjectTemplate
   Handle hoNETObjectTemplate
```

```
  Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate
  Set pvComObject of hoNETObjectTemplate to voNETObjectTemplate
    Send ComSetTemplateDef of hoNETObjectTemplate "Dim x"
    Send ComSetTemplateDef of hoNETObjectTemplate "1/1/2001"
    Set ComTemplate of hoNETObjectTemplate to "MaxSelectionCount =
1;SelectionStart = x"
  Send Destroy to hoNETObjectTemplate
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETObjectTemplate
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:BackgroundColor := 16777215

oNETHost:Create("C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__

      oNETObjectTemplate := oNETHost:Host()
         oNETObjectTemplate:SetTemplateDef("Dim x")
         oNETObjectTemplate:SetTemplateDef("01/01/2001")
```

```
        oNETObjectTemplate:Template := "MaxSelectionCount = 1;SelectionStart = x"

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

method NETObjectTemplate.SetValue (Value as Variant)

Specifies the value of the object.

Type	Description
Value as Variant	A VARIANT expression that specifies the new value to be assigned to the NETObjectTemplate object.

Use the SetValue property to change the object being hosted by the current NETObjectTemplate object. The Value property holds the original object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.Template as String

Executes the x-script code.

Type	Description
String	A String expression that specifies the x-script/template code to be executed.

Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. Prior to Template/Item call, you can invoke the SetTemplateDef to define values from your code to Template's code (TemplateDef variables).

in VB/NET under the .NET Framework, you use a code like follows to add nodes to a TreeView control:

```
With TreeView1
   With .Nodes.Add("Root 1")
      .Nodes.Add("Child 1")
      With .Nodes.Add("Child 2")
         .Nodes.Add("Sub-Child 2.1")
         .Nodes.Add("Sub-Child 2.2")
         .Nodes.Add("Sub-Child 2.3")
         .Expand()
      End With
      .Nodes.Add("Child 3")
      .Expand()
   End With
   With .Nodes.Add("Root 2")
      .Nodes.Add("Child 1")
      .Nodes.Add("Child 2")
      .Nodes.Add("Child 3")
   End With
End With
```

while on VB using the NETHost control you should use a code like:

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e089\Sy

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      With .Item("Nodes.Add(`Root 1`)")
         .Template = "Nodes.Add(`Child 1`)"
         With .Item("Nodes.Add(`Child 2`)")
            .Template = "Nodes.Add(`Sub-Child 2.1`)"
            .Template = "Nodes.Add(`Sub-Child 2.2`)"
            .Template = "Nodes.Add(`Sub-Child 2.3`)"
            .Template = "Expand()"
         End With
         .Template = "Nodes.Add(`Child 3`)"
         .Template = "Expand()"
      End With
      With .Item("Nodes.Add(`Root 2`)")
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Nodes.Add(`Child 3`)"
      End With
   End With
End With
```

The Template/ x-script code is a simple way of calling hosting control/object's properties, methods/ events using strings. Exontrol owns the x-script implementation in its easiest way and it does not require any VB engine to get executed. Our simple rule is using the component alone without any other dependency than the Windows system.

The **Template/x-script** syntax in BNF notation is defined like follows:

```
<x-script> := <lines>
<lines> := <line>[<eol> <lines>] | <block>
<block> := <call> [<eol>] { [<eol>] <lines> [<eol>] } [<eol>]
<eol> := ";" | "\r\n"
<line> := <dim> | <createobject> | <call> | <set> | <comment>
<dim> := "DIM" <variables>
```

```
<variables> := <variable> [, <variables>]
<variable> := "ME" | <identifier>
<createobject> := "CREATEOBJECT(`"<type>"`)"
<call> := <variable> | <property> | <variable>"."<property> | <createobject>"."
<property>
<property> := [<property>"."]<identifier>["("<parameters>")"]
<set> := <call> "=" <value>
<property> := <identifier> | <identifier>"("[<parameters>]")"
<parameters> := <value> [","<parameters>]
<value> := <boolean> | <number> | <color> | <date> | <string> | <createobject> |
<call>
<boolean> := "TRUE" | "FALSE"
<number> := "0X"<hexa> | ["-"]<integer>["."<integer>]
<digit10> := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<digit16> := <digit10> | A | B | C | D | E | F
<integer> := <digit10>[<integer>]
<hexa> := <digit16>[<hexa>]
<color> := "RGB("<integer>","<integer>","<integer>")"
<date> := "#"<integer>"/"<integer>"/"<integer>" "[<integer>":"<integer>":"
<integer>"]"#"
<string> := '"'<text>'"' | "`"<text>"`"
<comment> := "'"<text>
```

where:

<identifier> indicates an identifier of the variable, property or method, and should start with a letter.
<type> indicates the type the CreateObject function creates, as the assembly-qualified name of the type to create.
<text> any string of characters

The Template / x-script is composed by lines of instructions. Instructions are separated by "\r\n" (new line characters) or ";" character. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails. The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object.

An x-script instruction/line can be one of the following:

- **Dim variable**[, variable, ...] *declares the variables in the context. Multiple variables are separated by commas. The SetTemplateDef method can declare new variables to be available for the main context. (Sample: Dim h, h1, h2)*
- **variable =** [object.][property/method(arguments).]**property/method(arguments)** *assigns the result of the **property/method** call to the **variable**. (Sample: h = Nodes.Add(`Node`))*
- [object.][property/method(arguments).]**property(arguments) = value** *assigns the **value** to the **property.** (Sample: Nodes.Add(`Node`).BackColor = RGB(255,0,0))*
- [object.][property/method(arguments).]**property/method(arguments)** *invokes the **property/method**. (Sample: Nodes.Add(`Node`))*
- **{**context **}** *delimits the object's context. The properties/fields or methods called between { and } are related to the last object returned by the property/method prior to { declaration. (Sample: Nodes{Add(`Child 1`);Add(`Child 2`)})*
- ***.*** *delimits the object than its property or method. (Sample: Nodes.Add(`Element`), or Nodes.Add(`Element`) and Nodes{Add(`Element`)} are equivalents)*

where

- ***variable*** *is the name of a variable declared with Dim command or previously defined using the SetTemplateDef method.*
- ***property*** *is the name of a property/field of the current object in the current context.*
- ***method*** *is the name of a method of the current object in the current context.*
- ***arguments*** *include constants and/or variables and/or property/method calls separated by comma character.*
- ***object*** *can be a variable of an Object type, **Me** or **CreateObject** call.*

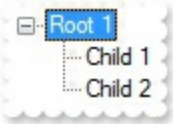
The x-script uses constant expressions as follows:

- *boolean* expression with possible values as ***True*** or ***False****. The True value is equivalent with -1, while False with 0. (Sample: Visible = False)*
- *numeric* expression may starts with **0x** which indicates a hexa decimal representation, else it should starts with digit, or +/- followed by a digit, and . is the decimal separator. *Sample: 13 indicates the integer 13, or **12.45** indicates the double expression 12,45. (Sample: BackColor = 0xFF0000)*
- *date* expression is delimited by **#** character in the format **#mm/dd/yyyy hh:mm:ss#**. *For instance, #31/12/1971# indicates the December 31, 1971 (Sample: Chart.FirstVisibleDate = #1/1/2001#)*
- *string* expression is delimited by **"** or ` characters. If using the ` character, please make sure that it is different than ' which allows adding comments inline. *Sample: **"text"** or **`text`** indicates the string text, while the ' text , specifies the comment text. (Sample: Text = "caption")*

Also , the template or x-script code supports general functions as follows:

- **Me** *property indicates the original object, and it is defined as a predefined variable. (Sample: Me.Nodes.Add(`Root 1`))*
- **RGB(**R,G,B**)** *property retrieves an RGB value, where the R, G, B are byte values that indicates the Red Green Blue bytes for the color being specified. (Sample: Nodes.Add(`Root 1`).BackColor = RGB(255,0,0))*
- **LoadPicture(**file**)** *property loads a picture from a file and returns a Picture object required by the picture properties. (Sample: BackgroundImage = LoadPicture(`C:\exontrol\images\auction.gif`)*
- **CreateObject(***assemblyQualifiedName***)** *property creates an instance of the specified type using that type's default constructor. The assemblyQualifiedName indicates the assembly-qualified name of the type to get. See AssemblyQualifiedName. If the type is in the currently executing assembly or in Mscorlib.dll, it is sufficient to supply the type name qualified by its namespace. (Sample: "CreateObject(`System.Windows.Forms.TabPage, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089`){Text = `Page`;UseVisualStyleBackColor = True}")*

The following samples shows how you can use the Template and TemplateResult properties:



VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`)"
      With .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      End With
   End With
```

```
End With
```

VB6

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`)"
      With .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      End With
   End With
End With
```

VB.NET

```
With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`)"
      With .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      End With
   End With
End With
```

VB.NET for /COM

```
With AxNETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`)"
      With .TemplateResult
         .Template = "Nodes.Add(`Child 1`)"
         .Template = "Nodes.Add(`Child 2`)"
         .Template = "Expand()"
      End With
   End With
End With
```

C++

```
/*
   Copy and paste the following directives to your header file as
   it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

   #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate = spNETHost1-
>GetHost();
   var_NETObjectTemplate->PutTemplate(L"Nodes.Add(`Root 1`)");
   exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate1 =
var_NETObjectTemplate->GetTemplateResult();
      var_NETObjectTemplate1->PutTemplate(L"Nodes.Add(`Child 1`)");
      var_NETObjectTemplate1->PutTemplate(L"Nodes.Add(`Child 2`)");
```

```
    var_NETObjectTemplate1->PutTemplate(L"Expand()");
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate = NETHost1-
>Host;
  var_NETObjectTemplate->Template = L"Nodes.Add(`Root 1`)";
  Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate1 =
var_NETObjectTemplate->TemplateResult;
    var_NETObjectTemplate1->Template = L"Nodes.Add(`Child 1`)";
    var_NETObjectTemplate1->Template = L"Nodes.Add(`Child 2`)";
    var_NETObjectTemplate1->Template = L"Expand()";
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = exnethost1.Host;
  var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)";
  exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate1 =
var_NETObjectTemplate.TemplateResult;
    var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)";
    var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)";
    var_NETObjectTemplate1.Template = "Expand()";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
```

```
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETObjectTemplate = NETHost1.Host;
      var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)";
      var var_NETObjectTemplate1 = var_NETObjectTemplate.TemplateResult;
         var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)";
         var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)";
         var_NETObjectTemplate1.Template = "Expand()";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1">></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
      With .Host
         .Template = "Nodes.Add(`Root 1`)"
         With .TemplateResult
            .Template = "Nodes.Add(`Child 1`)"
```

```
            .Template = "Nodes.Add(`Child 2`)"
            .Template = "Expand()"
        End With
    End With
  End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = axNETHost1.Host;
  var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)";
  exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate1 =
var_NETObjectTemplate.TemplateResult;
    var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)";
    var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)";
    var_NETObjectTemplate1.Template = "Expand()";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
  COM com_NETObjectTemplate,com_NETObjectTemplate1;
  anytype var_NETObjectTemplate,var_NETObjectTemplate1;
  ;

  super();



exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows
```

```
   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETObjectTemplate = exnethost1.Host(); com_NETObjectTemplate =
var_NETObjectTemplate;
      com_NETObjectTemplate.Template("Nodes.Add(`Root 1`)");
      var_NETObjectTemplate1 = com_NETObjectTemplate.TemplateResult();
com_NETObjectTemplate1 = var_NETObjectTemplate1;
         com_NETObjectTemplate1.Template("Nodes.Add(`Child 1`)");
         com_NETObjectTemplate1.Template("Nodes.Add(`Child 2`)");
         com_NETObjectTemplate1.Template("Expand()");
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'Nodes.Add(`Root 1`)';
      with TemplateResult do
      begin
         Template := 'Nodes.Add(`Child 1`)';
         Template := 'Nodes.Add(`Child 2`)';
         Template := 'Expand()';
      end;
   end;
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
  AssemblyName := 'System.Windows.Forms.TreeView';
  with Host do
  begin
    Template := 'Nodes.Add(`Root 1`)';
    with TemplateResult do
    begin
      Template := 'Nodes.Add(`Child 1`)';
      Template := 'Nodes.Add(`Child 2`)';
      Template := 'Expand()';
    end;
  end;
end
```

VFP

```
with thisform.NETHost1
  .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

  .AssemblyName = "System.Windows.Forms.TreeView"
  with .Host
    .Template = "Nodes.Add(`Root 1`)"
    with .TemplateResult
      .Template = "Nodes.Add(`Child 1`)"
      .Template = "Nodes.Add(`Child 2`)"
      .Template = "Expand()"
    endwith
  endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETObjectTemplate,var_NETObjectTemplate1

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)"
   var_NETObjectTemplate1 = var_NETObjectTemplate.TemplateResult
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)"
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)"
      var_NETObjectTemplate1.Template = "Expand()"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETObjectTemplate as P
Dim var_NETObjectTemplate1 as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)"
   var_NETObjectTemplate1 = var_NETObjectTemplate.TemplateResult
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)"
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)"
      var_NETObjectTemplate1.Template = "Expand()"
```

Visual Objects

```
local var_NETObjectTemplate as INETObjectTemplate
local var_NETObjectTemplate1 as INETObjectTemplate

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETObjectTemplate := oDCOCX_Exontrol1:Host
```

```
var_NETObjectTemplate:Template := "Nodes.Add(`Root 1`)"
var_NETObjectTemplate1 := var_NETObjectTemplate:TemplateResult
   var_NETObjectTemplate1:Template := "Nodes.Add(`Child 1`)"
   var_NETObjectTemplate1:Template := "Nodes.Add(`Child 2`)"
   var_NETObjectTemplate1:Template := "Expand()"
```

PowerBuilder

```
OleObject oNETHost,var_NETObjectTemplate,var_NETObjectTemplate1

oNETHost = ole_1.Object
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`)"
   var_NETObjectTemplate1 = var_NETObjectTemplate.TemplateResult
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 1`)"
      var_NETObjectTemplate1.Template = "Nodes.Add(`Child 2`)"
      var_NETObjectTemplate1.Template = "Expand()"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETObjectTemplate
   Get ComHost to voNETObjectTemplate
   Handle hoNETObjectTemplate
   Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate
   Set pvComObject of hoNETObjectTemplate to voNETObjectTemplate
      Set ComTemplate of hoNETObjectTemplate to "Nodes.Add(`Root 1`)"
```

```
    Variant voNETObjectTemplate1
    Get ComTemplateResult of hoNETObjectTemplate to voNETObjectTemplate1
    Handle hoNETObjectTemplate1
    Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate1
    Set pvComObject of hoNETObjectTemplate1 to voNETObjectTemplate1
      Set ComTemplate of hoNETObjectTemplate1 to "Nodes.Add(`Child 1`)"
      Set ComTemplate of hoNETObjectTemplate1 to "Nodes.Add(`Child 2`)"
      Set ComTemplate of hoNETObjectTemplate1 to "Expand()"
    Send Destroy to hoNETObjectTemplate1
  Send Destroy to hoNETObjectTemplate
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETObjectTemplate
   LOCAL oNETObjectTemplate1
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08
```

```
      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETObjectTemplate := oNETHost:Host()
         oNETObjectTemplate:Template := "Nodes.Add(`Root 1`)"
         oNETObjectTemplate1 := oNETObjectTemplate:TemplateResult()
            oNETObjectTemplate1:Template := "Nodes.Add(`Child 1`)"
            oNETObjectTemplate1:Template := "Nodes.Add(`Child 2`)"
            oNETObjectTemplate1:Template := "Expand()"

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

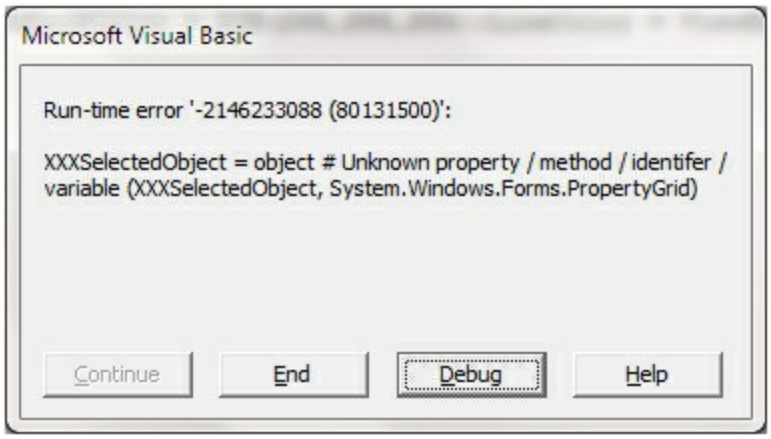
property NETObjectTemplate.TemplateError as Long

Indicates the error code of the last Template call.

Type	Description
Long	A long expression that describes the error/exception in the Item/Template call.

By default, the TemplateError property returns 0. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

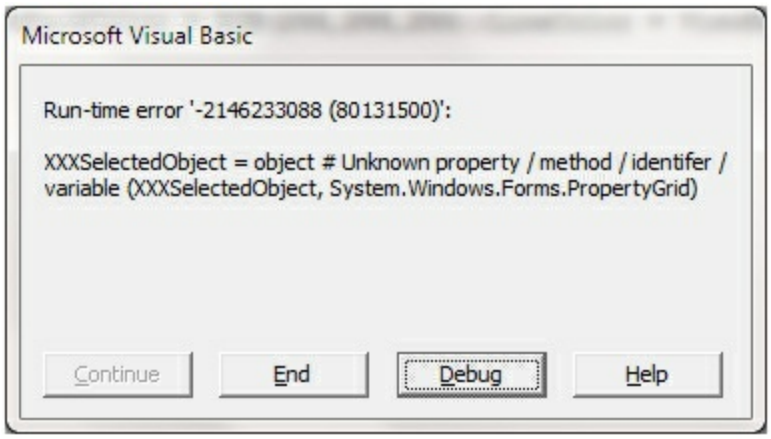
property NETObjectTemplate.TemplateException as String

Indicates the detailed information about the exception that occurs.

Type	Description
String	A String expression that describes the error/exception in the Item/Template call.

By default, the TemplateException property is empty. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

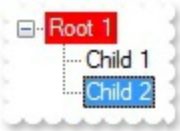
property NETObjectTemplate.TemplateResult as NETObjectTemplate

Indicates the result of the last Template call.

Type	Description
NETObjectTemplate	A NETObjectTemplate object that holds the result of the last Template call.

The TemplateResult property returns the result of the last instruction into a Template call, as a NETObjectTemplate object. Use the Template/Item property to get/set properties / fields / parameters, invoke methods of the hosting /NET framework Value, using the x-script code. The Item property does exactly the same thing as Template call, excepts that it returns the TemplateResult property. For instance, using the Template/Item property you can change the hosting control's background color, add nodes, and so on. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException gets the error if the Template calls fails.

The following samples shows how you can use the Template and TemplateResult properties:



How can I use the TemplateResult property?

VBA (MS Access, Excell...)

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB6

```
With NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB.NET

```
With Exnethost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
End With
```

VB.NET for /COM

```
With AxNETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   With .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   End With
```

```
End With
```

C++

```
/*
   Copy and paste the following directives to your header file as
   it defines the namespace 'exontrol_NETHost' for the library: 'Exontrol NETHost
ActiveX Component'

   #import <exontrol.NETHost.tlb>
*/
exontrol_NETHost::INETHostCtrlPtr spNETHost1 = GetDlgItem(IDC_NETHOST1)-
>GetControlUnknown();
spNETHost1-
>PutAssemblyLocation(L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Form

spNETHost1->PutAssemblyName(L"System.Windows.Forms.TreeView");
exontrol_NETHost::INETObjectTemplatePtr var_NETObjectTemplate = spNETHost1-
>GetHost();
   var_NETObjectTemplate->PutTemplate(L"Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }");
   var_NETObjectTemplate->GetTemplateResult()->PutTemplate(L"Nodes{
Add(`Child 1`); Add(`Child 2`) }; Expand() }");
```

C++ Builder

```
NETHost1->AssemblyLocation =
L"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c5619

NETHost1->AssemblyName = L"System.Windows.Forms.TreeView";
Exontrol_nethost_tlb::INETObjectTemplatePtr var_NETObjectTemplate = NETHost1-
>Host;
   var_NETObjectTemplate->Template = L"Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETObjectTemplate->TemplateResult->Template = L"Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

C#

```
exnethost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

exnethost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = exnethost1.Host;
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

JScript/JavaScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="JScript">
function Init()
{
   NETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

   NETHost1.AssemblyName = "System.Windows.Forms.TreeView";
   var var_NETObjectTemplate = NETHost1.Host;
      var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
      var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
}
</SCRIPT>
</BODY>
```

VBScript

```
<BODY onload="Init()">
<OBJECT CLASSID="clsid:FDCBA3E0-4E2F-4DC7-B073-EAA7BD7EC565"
id="NETHost1"></OBJECT>

<SCRIPT LANGUAGE="VBScript">
Function Init()
   With NETHost1
      .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      .AssemblyName = "System.Windows.Forms.TreeView"
      With .Host
         .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
         .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand()
}"
      End With
   End With
End Function
</SCRIPT>
</BODY>
```

C# for /COM

```
axNETHost1.AssemblyLocation =
"C:\\Windows\\assembly\\GAC_MSIL\\System.Windows.Forms\\2.0.0.0__b77a5c56193

axNETHost1.AssemblyName = "System.Windows.Forms.TreeView";
exontrol_NETHost.NETObjectTemplate var_NETObjectTemplate = axNETHost1.Host;
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }";
   var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }";
```

X++ (Dynamics Ax 2009)

```
public void init()
{
   COM com_NETObjectTemplate,com_NETObjectTemplate1;
   anytype var_NETObjectTemplate,var_NETObjectTemplate1;
   ;

   super();



exnethost1.AssemblyLocation("C:\\Windows\\assembly\\GAC_MSIL\\System.Windows

   exnethost1.AssemblyName("System.Windows.Forms.TreeView");
   var_NETObjectTemplate = exnethost1.Host(); com_NETObjectTemplate =
var_NETObjectTemplate;
      com_NETObjectTemplate.Template("Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }");
      var_NETObjectTemplate1 =
COM::createFromObject(com_NETObjectTemplate.TemplateResult());
com_NETObjectTemplate1 = var_NETObjectTemplate1;
      com_NETObjectTemplate1.Template("Nodes{ Add(`Child 1`); Add(`Child 2`) };
Expand() }");
}
```

Delphi 8 (.NET only)

```
with AxNETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }';
      TemplateResult.Template := 'Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }';
   end;
```

```
end
```

Delphi (standard)

```
with NETHost1 do
begin
   AssemblyLocation :=
'C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   AssemblyName := 'System.Windows.Forms.TreeView';
   with Host do
   begin
      Template := 'Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }';
      TemplateResult.Template := 'Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }';
   end;
end
```

VFP

```
with thisform.NETHost1
   .AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   .AssemblyName = "System.Windows.Forms.TreeView"
   with .Host
      .Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor =
RGB(255,255,255) }"
      .TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
   endwith
endwith
```

dBASE Plus

```
local oNETHost,var_NETObjectTemplate

oNETHost = form.Activex1.nativeObject
oNETHost.AssemblyLocation =
```

```
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
```

XBasic (Alpha Five)

```
Dim oNETHost as P
Dim var_NETObjectTemplate as P

oNETHost = topparent:CONTROL_ACTIVEX1.activex
oNETHost.AssemblyLocation =
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
```

Visual Objects

```
local var_NETObjectTemplate as INETObjectTemplate

oDCOCX_Exontrol1:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oDCOCX_Exontrol1:AssemblyName := "System.Windows.Forms.TreeView"
var_NETObjectTemplate := oDCOCX_Exontrol1:Host
   var_NETObjectTemplate:Template := "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
```

```
var_NETObjectTemplate:TemplateResult:Template := "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
```

PowerBuilder

```
OleObject oNETHost,var_NETObjectTemplate

oNETHost = ole_1.Object
oNETHost.AssemblyLocation = "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

oNETHost.AssemblyName = "System.Windows.Forms.TreeView"
var_NETObjectTemplate = oNETHost.Host
   var_NETObjectTemplate.Template = "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor = RGB(255,255,255) }"
   var_NETObjectTemplate.TemplateResult.Template = "Nodes{ Add(`Child 1`); Add(`Child 2`) }; Expand() }"
```

Visual DataFlex

```
Procedure OnCreate
   Forward Send OnCreate
   Set ComAssemblyLocation to "C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

   Set ComAssemblyName to "System.Windows.Forms.TreeView"
   Variant voNETObjectTemplate
   Get ComHost to voNETObjectTemplate
   Handle hoNETObjectTemplate
   Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate
   Set pvComObject of hoNETObjectTemplate to voNETObjectTemplate
      Set ComTemplate of hoNETObjectTemplate to "Nodes.Add(`Root 1`){ BackColor = RGB(255,0,0);ForeColor = RGB(255,255,255) }"
      Variant voNETObjectTemplate1
      Get ComTemplateResult of hoNETObjectTemplate to voNETObjectTemplate1
      Handle hoNETObjectTemplate1
```

```
      Get Create (RefClass(cComNETObjectTemplate)) to hoNETObjectTemplate1
      Set pvComObject of hoNETObjectTemplate1 to voNETObjectTemplate1
         Set ComTemplate of hoNETObjectTemplate1 to "Nodes{ Add(`Child 1`);
Add(`Child 2`) }; Expand() }"
      Send Destroy to hoNETObjectTemplate1
   Send Destroy to hoNETObjectTemplate
End_Procedure
```

XBase++

```
#include "AppEvent.ch"
#include "ActiveX.ch"

PROCEDURE Main
   LOCAL oForm
   LOCAL nEvent := 0, mp1 := NIL, mp2 := NIL, oXbp := NIL
   LOCAL oNETObjectTemplate
   LOCAL oNETHost

   oForm := XbpDialog():new( AppDesktop() )
   oForm:drawingArea:clipChildren := .T.
   oForm:create( ,,{100,100}, {640,480},, .F. )
   oForm:close  := {|| PostAppEvent( xbeP_Quit )}

   oNETHost := XbpActiveXControl():new( oForm:drawingArea )
   oNETHost:CLSID  := "Exontrol.NETHost" /*{FDCBA3E0-4E2F-4DC7-B073-
EAA7BD7EC565}*/
   oNETHost:create(,, {10,60},{610,370} )

      oNETHost:AssemblyLocation :=
"C:\Windows\assembly\GAC_MSIL\System.Windows.Forms\2.0.0.0__b77a5c561934e08

      oNETHost:AssemblyName := "System.Windows.Forms.TreeView"
      oNETObjectTemplate := oNETHost:Host()
         oNETObjectTemplate:Template := "Nodes.Add(`Root 1`){ BackColor =
RGB(255,0,0);ForeColor = RGB(255,255,255) }"
         oNETObjectTemplate:TemplateResult():Template := "Nodes{ Add(`Child 1`);
```

```
Add(`Child 2`) }; Expand() }"

   oForm:Show()
   DO WHILE nEvent != xbeP_Quit
      nEvent := AppEvent( @mp1, @mp2, @oXbp )
      oXbp:handleEvent( nEvent, mp1, mp2 )
   ENDDO
RETURN
```

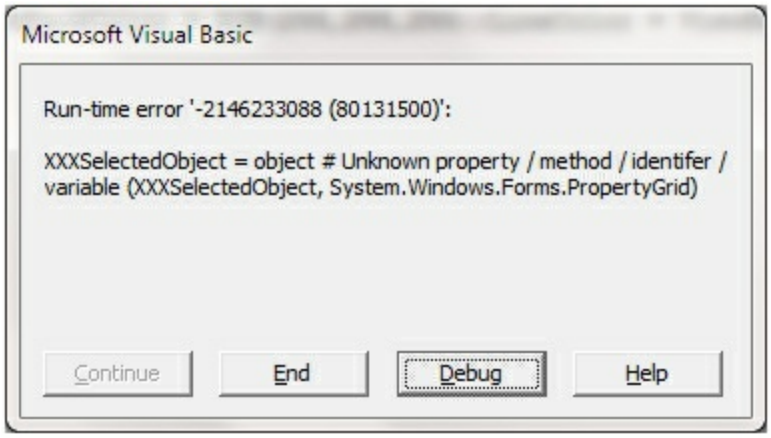
property NETObjectTemplate.TemplateThrowError as Boolean

Specifies whether the execution of the template stops once an error occurs.

Type	Description
Boolean	A Boolean expression that specifies whether the NETHost control fires an error/exception when an error occurs in the Item/Template call.

By default, the TemplateThrowError property is True. The TemplateThrowError property specifies whether the control fires an exception/error when the Template call fails. The TemplateError / TemplateException property indicates the error/exception that occurred in the Item/Template call. The TemplateError / TemplateException gets the error if the Template calls fails.

By default, the control fires an exception/error when the Item/Template call fails like shown in the following screen shot:



while running the following code:

```
Private Sub Form_Load()
   With NETHost1
      .AssemblyQualifiedName = "System.Windows.Forms.PropertyGrid,
System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089"
      With .Host
         .Template = "BackColor = RGB(255,255,255);ViewBackColor =
RGB(255,255,255);LineColor = ViewBackColor"
         .SetTemplateDef "dim object"
         .SetTemplateDef Me
         .Template = "XXXSelectedObject = object"
      End With
```

```
    End With
End Sub
```

where intentionally we used ~~XXX~~SelectedObject instead of SelectedObject property of the /NET Framework's PropertyGrid.

If the TemplateThrowError property is False (by default, it is True), you can get the error/exception using the TemplateError / TemplateException property as:

```
TemplateError is -2147352570
TemplateException is XXXSelectedObject = object # Unknown property / method / identifer / variable (XXXSelectedObject, System.Windows.Forms.PropertyGrid)
```

property NETObjectTemplate.Type as String

Indicates the type of the object's value.

Type	Description
String	A string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. For instance: "System.Windows.Forms.TreeView, System.Windows.Forms, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089"

property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property. The Value property holds the original object.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.Value as Variant

Specifies the value of the object.

Type	Description
Variant	A VARIANT expression that specifies the original object (.NET Framework object) hold by the current NETObjectTemplate object.

The Value property holds the original object. Use the SetValue property to change the object being hosted by the current NETHostObject object. The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.

property NETObjectTemplate.VtType as NETHostVarEnum

Indicates the type/vartype of the object's value.

Type	Description
NETHostVarEnum	A NETHostVarEnum expression that specifies the VARIANT-type of the Value.

The VtType property indicates the VARIANT type of the object that the current NETObjectTemplate object holds. The Type property returns a string that specifies the fully assembly-qualified name of the type, which includes the name of the assembly from which this Type object is loaded. If the NETObjectTemplate holds a class or an object/IDispatch/IUnknown that supports properties, fields, members, any of these can be called through the NETObjectTemplate properties like: Item, SetTemplateDef or Template property. The Value property holds the original object.

You can use the following properties to convert the current Value to indicated standard types:

- AsBoolean, converts the value to a boolean expression.
- AsDate, converts the value to a DATE-TIME/double expression.
- AsDouble, converts the value to a double expression.
- AsInt, converts the value to an integer-32 expression.
- AsString, gets the value converted to a string expression.