IRCommand2 COM Automation Interface
Table Of Contents

Section 1 Introduction / Overview 1-1
1.1 Purpose 1-1
1.2 Features Overview 1-1
1.3 The COM Object Hierarchy 1-2

Section 2 IRCommand2 COM Object Classes 2-1
2.1 IRC2Application 2-1
2.2 IRC2DeviceCollection 2-1
2.3 IRC2Device 2-1
2.4 IRC2ButtonCollection 2-1
2.5 IRC2Button 2-2
2.6 IRC2ButtonGroup 2-2

Section 3 IRCommand2 COM Object Interface Reference 3-1
3.1 IRC2Application 3-1
3.1.1 Properties 3-1
3.1.1.1 DeviceCount As Variant 3-1
3.1.1.2 Devices As Object 3-1
3.1.2 Methods 3-1
3.1.2.1 Click 3-1
3.1.2.2 GetClick 3-1
3.1.2.3 ClickButton (Rev 3.80 or later) 3-1
3.1.2.4 FindButton (Rev 3.80 or later) 3-2
3.1.2.5 ClickNumber (Rev 3.80 or later) 3-2
3.2 IRC2DeviceCollection 3-3
3.2.1 Properties 3-3
3.2.1.1 Property Item(Index As Integer) As Object 3-3
3.2.2 Methods 3-3
3.3 IRC2Device 3-3
3.3.1 Properties 3-3
3.3.1.1 ButtonCount As Integer 3-3
3.3.1.2 Buttons As Object 3-3
3.3.1.3 Name As Variant 3-3
3.3.1.4 Type As Integer 3-4
3.3.1.5 Color(ColorItem As Integer) As Long (Rev 3.89 or later) 3-4
3.3.1.6 IsHidden As BOOL (Rev 3.89 or later) 3-4
3.3.1.7 PosSize As Variant Array (Rev 5.76 or later) 3-4
3.3.2 Methods 3-4
3.4 IRC2ButtonCollection 3-4
3.4.1 Properties 3-5
3.4.1.1 Property Item(Index As Integer) As Object 3-5
3.4.2 Methods 3-5
3.5 IRC2Button 3-5

3.5.1 Properties 3-5
3.5.1.1 GrpButtons As Object 3-5
3.5.1.2 GrpCount As Integer 3-5
3.5.1.3 IconIndex As Integer (Rev 3.69 or later) 3-5
3.5.1.4 IconSize As Integer (Rev 3.73 or later) 3-5
3.5.1.5 Name As Variant 3-6
3.5.1.6 Parent As Object 3-6
3.5.1.7 PathName As Variant 3-6
3.5.1.8 PosSize As Variant Array (Rev 3.69 or later) 3-6
3.5.1.9 ShortName As Variant (Rev 3.73 or later) 3-6
3.5.1.10 Type As Integer 3-7
3.5.1.11 Color(ColorItem As Integer) As Long (Rev 3.89 or later) 3-7
3.5.2 Methods 3-7
3.5.2.1 RegTrigger 3-7
3.6 IRC2ButtonGroup 3-7
3.6.1 Properties 3-7
3.6.1.1 Property Item(Index As Integer) As Object 3-7
3.6.1.2 Button Properties 3-8
3.6.2 Methods 3-8

Section 4 Sample Code 4-1
4.1 Enumerating Objects 4-1
4.2 Getting a Button Selection 4-2
4.3 Setting a Cable Channel 4-2

Section 1
Introduction / Overview

1.1 Purpose

IRCommand2 provides COM (Component Object Model) Automation services that enable client applications to directly control any device that has been configured for control within IRCommand2[1]. This powerful capability makes it possible for any application to control devices without the complexity of interfacing directly with hardware. Applications written in virtually any language that supports COM can take advantage of this capability.

This important feature of IRCommand2 is also compatible with DCOM (Distributed COM), which enables the client application and the server (IRCommand2) to reside on different computers on a network.

1.2 Features Overview

The IRCommand2 COM Automation interface enables a client application to:

a. Obtain a persistent tag to any defined button within IRCommand2. This tag can be obtained by searching all defined buttons for a specific name, or by requesting that IRCommand2 prompt the user to click on a specific button. Button tags are "persistent" in that they remain valid between sessions, as long as the button is not deleted

b. Obtain names, counts, and other relevant properties of device panels and buttons

c. Perform a "click" of a button referenced by its persistent tag

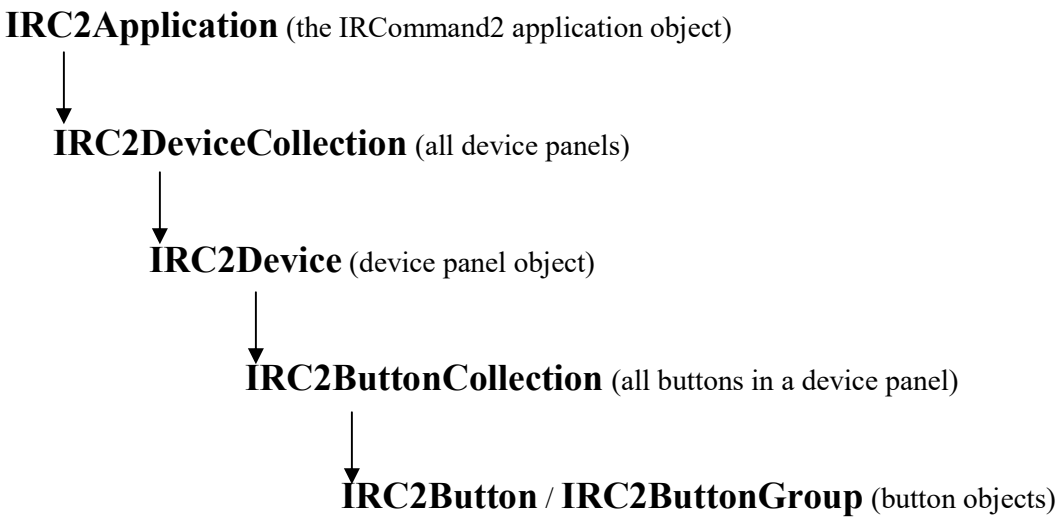
d. Click or find a button by name (new in Rev 3.80)

e. Click a string of numbers using a single function call (new in Rev 3.80).

[1] You must register IRCommand2 in order to use the COM interface; it is not available in the demo mode.

1.3 The COM Object Hierarchy

The IRCommand2 COM Automation interface consists of a hierarchy of interface objects whose classes are defined as follows:



Each of the above object classes supports specific properties and methods, as defined in the sections that follow.

Section 2
IRCommand2 COM Object Classes

The IRCommand2 COM Automation interface enables interaction with all object classes. These classes are not normally referenced directly, since it's really the interface to each class that is of interest. However, if the type library for IRCommand2 is examined with an object browser, the following object classes will appear[2].

2.1 IRC2Application

The IRC2Application class represents the application object. Creation of this object establishes the link to the IRCommand2 server. When creating this object via COM you can reference the name "IRCommand2.App" to create the IRCommand2 application object. If IRCommand2 is not already running then it will be started automatically by COM[3].

2.2 IRC2DeviceCollection

The IRC2DeviceCollection class represents the full collection of device panels contained within the IRCommand2 application. By using this class, you can enumerate all of the device panels and obtain an IRC2Device object reference to each device panel.

2.3 IRC2Device

The IRC2Device class represents a specific device panel object within IRCommand2. An object of class IRC2Device is necessary in order to gain access to the buttons contained within the associated device panel and to access other device panel properties, such as the device panel name. You can obtain a reference to an IRC2Device object via the IRC2DeviceCollection object.

2.4 IRC2ButtonCollection

The IRC2ButtonCollection class represents the full collection of buttons contained within a device panel object. By using this class, you can enumerate all of the buttons and obtain an IRC2Button object reference to each button.

[2] Make sure you're using a version of the type library dated 11/20/03 or later. Earlier versions had some of the object classes mis-named.

[3] For COM, startup of IRCommand2 will always occur as long as the client and server are on the same computer. For DCOM, where the client and server are on different computers, IRCommand2 will start on the server only if it is running a Server edition of Windows. Otherwise, you must manually start IRCommand2 on the server before running remote clients.

2.5 IRC2Button

The IRC2Button class represents a specific button object within IRCommand2. An object of class IRC2Button is necessary in order to gain access to button properties, such as the button's name, type, and its persistent tag. You can obtain a reference to an IRC2Button object via the IRC2ButtonCollection object.

2.6 IRC2ButtonGroup

The IRC2ButtonGroup class represents a specific button group object defined within IRCommand2. Button groups are a special type of button. Any IRC2Button object of the "button group" type supports this additional class, which enables access to the individual buttons within the group, as well as to properties of the button group. One example of a button group is the numeric pad, which is a button group containing 10 buttons. State buttons are also treated as button groups.

Section 3
IRCommand2 COM Object Interface Reference

The following sections describe the interface for each COM object in IRComand2. The descriptions include full documentation for all supported properties and methods of each interface. Call syntax is shown in Visual Basic format.

3.1 IRC2Application

3.1.1 Properties

3.1.1.1 DeviceCount As Variant

A numeric value equal to the number of device panels defined in IRCommand2.

3.1.1.2 Devices As Object

An object of type IRC2DeviceCollection that can be used to enumerate and examine all device objects.

3.1.2 Methods

3.1.2.1 Click

Click (XTrigger As String) As Boolean

This function is used to click a button. The input parameter Xtrigger is a string containing the button's persistent tag. The function returns TRUE if successful, otherwise FALSE.

3.1.2.2 GetClick

GetClick (sMsg As String) As Object

This function causes IRCommand2 to prompt the user to click a button. The input parameter is a string that you want to display to the user to explain what to click on. If the user clicks on a button, an object of type IRC2Button is returned. If the user cancels the operation instead clicking on a button then a NULL object is returned. This function can also throw an exception error (trapped by On Error in VB) if a failure condition is encountered.

3.1.2.3 ClickButton (Rev 3.80 or later)

ClickButton (Device As String, BtnPath As String) As Boolean

This function is used to click a button by name. The input parameters are defined as follows:

Device: the name of the device panel containing the button to click.

BtnPath: the pathname to the button to click. If the button is not in a group then this is just the button name as defined by its label. If the button is in a group then the pathname consists of the group name and button name separated by a dash. For example, "Channel-0" to click the "0" button in the numeric pad labeled "Channel". Note: if the button label contains dashes ("-") then omit the dashes in this parameter.

The function returns TRUE if the button was found and clicked; otherwise it returns FALSE.

3.1.2.4 FindButton (Rev 3.80 or later)

FindButton(Device As String, BtnPath As String) As Object

This function is used to find a button by name and return an object reference to it. The input parameters are defined as follows:

Device: the name of the device panel containing the button to click.

BtnPath: the pathname to the button to click. If the button is not in a group then this is just the button name as defined by its label. If the button is in a group then the pathname consists of the group name and button name separated by a dash. For example, "Channel-0" to click the "0" button in the numeric pad labeled "Channel". Note: if the button label contains dashes ("-") then omit the dashes in this parameter.

The function returns a reference to the button object if the button was found; otherwise it returns NULL.

3.1.2.5 ClickNumber (Rev 3.80 or later)

ClickNumber(Device As String, GrpPath As String, Number As String) As Boolean

This function clicks a multi-digit number sequence. The input parameters are defined as follows:

Device: the name of the device panel containing the numeric buttons to click.

GrpPath: the pathname to the group containing the numeric buttons to click. Normally, this group would be a numeric pad with a label. Note: if the group label contains dashes ("-") then omit the dashes in this parameter.

Number: the number sequence to click. Each digit in the string is clicked in sequence from left to right. The buttons in the group to click must be labeled as single digit numbers, as would be the case for a numeric pad. The string is not restricted to numbers, but there must be a button in the group for each character in the string.

The function returns TRUE if at least one button was clicked; otherwise it returns FALSE.

3.2 IRC2DeviceCollection

This object enables access to individual device objects by specifying a numeric index to the property.

3.2.1 Properties

3.2.1.1 Property Item(Index As Integer) As Object

An object of type IRC2Device for the device at the specified index position. The valid index range is from one to the device count; otherwise, an “invalid argument” exception is thrown.

3.2.2 Methods

None.

3.3 IRC2Device

3.3.1 Properties

3.3.1.1 ButtonCount As Integer

A numeric value equal to the number of buttons contained within this device object. Note that each button group counts as one button in this count. In other words, it is a count of all "top level" buttons in the device.

3.3.1.2 Buttons As Object

An object of type IRC2ButtonCollection that can be used to enumerate and examine all button objects in this device.

3.3.1.3 Name As Variant

The name of this device object, as defined by the user within IRCommand2. The value is always a string.

3.3.1.4 Type As Integer

The numeric device type of this device object. At present, there is only one type of device, which has a device type = 10.

3.3.1.5 Color(ColorItem As Integer) As Long (Rev 3.89 or later)

The RGB value for a color property of this device. Valid ColorItem values are as follows:

1 = Fill Color
2 = Text Color.

3.3.1.6 IsHidden As BOOL (Rev 3.89 or later)

Property is TRUE if device is hidden from user view; otherwise FALSE.

3.3.1.7 PosSize As Variant Array (Rev 5.76 or later)

The size and position of the device panel within the program window. Elements of the returned variant array are as follows:

0 – position of left side of device
1 – position of top side of device
2 – Width of device
3 – Height of device.

Note that all Device panels currently have the same size and position. The primary value of this new property is to get overall dimensions for a Device panel. Device panel dimensions are determined by the chosen program window size and the aspect ratio for a specific configuration. Element values are CDLUs, which are DLUs (Device Logical Units) times 100 (to minimize roundoff errors). The base conversions to pixels are as follows:

x pixels = x CDLUs / 400
y pixels = y CDLUs / 800.

3.3.2 Methods

None.

3.4 IRC2ButtonCollection

This object enables access to individual button objects within a device by specifying a numeric index to the property.

3.4.1 Properties

3.4.1.1 Property Item(Index As Integer) As Object

An object of type IRC2Button for the button at the specified index position within this device object. The valid index range is from one to the button count for the device; otherwise, an "invalid argument" exception is thrown.

3.4.2 Methods

None.

3.5 IRC2Button

3.5.1 Properties

3.5.1.1 GrpButtons As Object

An object of type IRC2ButtonGroup that can be used to enumerate and examine the buttons contained within this button, which must be of type "button group". Value is NULL if this button is not a button group. An example of a button group is the numeric pad.

3.5.1.2 GrpCount As Integer

A numeric value equal to the number of buttons contained within this button, which is meaningful only for buttons of type "button group". Value is zero if this button is not a button group.

3.5.1.3 IconIndex As Integer (Rev 3.69 or later)

A numeric value for the index to the icon in this button object. Value is –1 if no icon is set.

3.5.1.4 IconSize As Integer (Rev 3.73 or later)

A numeric value for the size of the icon in this button object. The possible values are defined as follows:

0 = Small
1 = Medium
2 = Large
3 = Extra Large
-1 = Best Fit for icon size

This property has meaning only if **IconIndex** is greater than –1.

3.5.1.5 Name As Variant

The name of this button object, as defined by the user within IRCommand2. The value is always a string. If the button is within one or more groups then the button name returned is prefixed with the names of all containing groups in order from the outermost to innermost group with each group name separated by a dash "-" (e.g., "Power-0").

3.5.1.6 Parent As Object

An object of type IRC2ButtonGroup or IRC2Device that is the parent (i.e., contains) this button object. The type will be IRC2ButtonGroup if the button is contained in a button group; otherwise it will be of type IRC2Device. The “Type” property of the parent can be examined to determine the type.

3.5.1.7 PathName As Variant

The long name of this button object, which is more likely to be unique. This name consists of the containing device panel name followed by a slash "/" and then the button's Name property (e.g., TV/Power-0").

3.5.1.8 PosSize As Variant Array (Rev 3.69 or later)

The size and position within the device panel of this button object. Elements of the returned variant array are as follows:

0 – position of left side of button
1 – position of top side of button
2 – Width of button
3 – Height of button.

Element values are CDLUs, which are DLUs (Device Logical Units) times 100 (to minimize roundoff errors). The base conversions to pixels are as follows:

x pixels = x CDLUs / 400
y pixels = y CDLUs / 800.

3.5.1.9 ShortName As Variant (Rev 3.73 or later)

The name of this button object, as defined by the user within IRCommand2. The value is always a string. The difference between this property and the **Name** property is that this property does not include containing group names.

3.5.1.10 Type As Integer

The numeric type of this button object. Possible values are as follows:

0 = Button
1 = Button group
2 = State group.

3.5.1.11 Color(ColorItem As Integer) As Long (Rev 3.89 or later)

The RGB value for a color property of this button or group. Valid ColorItem values are as follows:

1 = Fill Color
2 = Text Color.

3.5.2 Methods

3.5.2.1 RegTrigger

RegTrigger() As Variant

Registers this button object for subsequent triggering via the IRC2Application Click method. This function is used to get the persistent tag for a button. The returned value is always a string representing a GUID. This method will return an error for button group objects since they cannot be clicked.

3.6 IRC2ButtonGroup

This object enables access to individual button objects within a button group by specifying a numeric index to the property.

3.6.1 Properties

3.6.1.1 Property Item(Index As Integer) As Object

An object of type IRC2Button for the button at the specified index position within the button group. The valid index range is from one to the button count for the button group; otherwise, an “invalid argument” exception is thrown. This property is not typically used explicitly. It exists to support For/Next object iterations in VB and in scripts.

3.6.1.2 Button Properties

Since a group is a type of button, it also has the properties of a button. See section 3.5 for information on button properties.

3.6.2 Methods

None.

Section 4
Sample Code

The following code segments help illustrate how to use the interface in Visual Basic. As might be expected, coding for the interface is a bit more complex in C or C++. Future versions of this document may include samples for C++.

4.1 Enumerating Objects

The following sample code creates an IRC2 application object and then walks through all of the buttons contained within the first device panel.

```
Dim IRC As Object
Set IRC = CreateObject("IRCommand2.App")
Dim x As Object, y As Object
Dim DevName as String
Dim ButtonCount as Integer
Dim RowText as String

DevName = IRC.Devices(0).Name
ButtonCount = IRC.Devices(0).ButtonCount
RowText = "Button Label, , Type, #Grp; "
For Each x In IRC.Devices(0).Buttons
   If x.Type > 0 Then ' Group button
      ' First list the parent group button
      RowText = RowText + x.Name + " ," + "," + Str(x.Type) + "," + Str(x.GrpCount) + ";"
      ' Then loop through the contained buttons
      For Each y In x.GrpButtons
         RowText = RowText + "------," + y.Name + "," + "," + ";"
      Next
   Else
      RowText = RowText + x.Name + " ," + "," + Str(x.Type) + "," + ";"
   End If
Next

Set IRC = Nothing
```

4.2 Getting a Button Selection

The following code creates an IRC2 application object and then fetches the total device count for the application. It then gets IRCommand2 to prompt the user to click on a button and that button is then registered for later triggering.

```
Dim vGuid As Variant
Dim Btn As Object
Dim DevCnt as Integer
Dim IRC As Object

Set IRC = CreateObject("IRCommand2.App")
DevCnt = IRC.DeviceCount
On Error GoTo NoButton
Set Btn = IRC.GetClick("Click a button to register")
vGuid = Btn.RegTrigger
On Error GoTo 0
.
.

NoButton:
.
.
.
.
```

4.3 Setting a Cable Channel

The following code illustrates how the new ClickNumber function can be used to click a sequence of numeric buttons to set a channel on a cable box. The device panel is named "Cable" and the numeric pad is labeled "Channel".

```
Dim IRC As Object
Dim bStatus As Boolean

Set IRC = CreateObject("IRCommand2.App")
bStatus = IRC.ClickNumber("Cable", "Channel", "145")
If bStatus Then
        MessageBox("Success!")
Else
        MessageBox("Failed")
End If
```