HouseBot ZonePlayer Device

Getting Started:

1. Move the WSC folder to C:\WSC. Navigate to ZonePlayerController.wsc and ZonePlayerFinder.wsc in the WSC folder, right-click each .wsc file, and select Register from the drop-down menu that appears.

2. Double-click UniqueDeviceNames.vbs in the Scripts folder to run the script. This will place a text file in C:\UniqueDeviceNames.txt that contains the zone name and unique device name of each of the ZonePlayers on your network. The unique device name (UDN) will be needed for the HouseBot device(s) to function properly.

3. If you only want to add a ZonePlayer device to HouseBot (i.e., no theme) then move a **COPY** of the Single ZonePlayer Device Import.hbx file in the HouseBot Import Files folder to C:\Program Files\HouseBot\Config\Import and restart HouseBot. This will add a single ZonePlayer device to your device list. Rename the device to reflect the name of the zone it is to control following the format ZP_<Zone Name> (**NOTE:** ZP_ must precede the zone name for all devices in order for the script to function correctly). Continue this procedure until you have added sufficient devices to HouseBot to reflect the zones on your Sonos network.

4. If you would like to add the 800x480 theme to HouseBot (includes 3 ZonePlayer devices: Family Room, Master Bedroom, Playroom as well as 3 Tasks) then move the Sonos 800 x 480 theme.hbx file in HouseBot Import Files to C:\Program Files\HouseBot\Config\Import and restart HouseBot. You must place the entire SonosHH Images folder in HouseBot Import Files in C:\Program Files\HouseBot\Config\Themes for the theme to work correctly.

5. Open C:\UniqueDeviceNames.txt and cut-and-paste the UDN for each zone to the UDN property of that device in HouseBot (select the device in the left pane and then navigate to UDN in the right pane and paste into the text box).

6. Next, ensure that the script device is pointing to the most current version of the script. Change the State property of the device to running. The current settings of that zone will populate the corresponding properties in the device.

The theme is included to provide a jumping-off point for you to create your own theme. It is more of a demonstration of how to use the device properties in conjunction with the available theme controls to provide a user-interface to interact with the Sonos system via HouseBot. The Sonos handheld controller is recapitulated in this theme to make it a bit easier to understand how the theme was constructed.

Below is a list of the device properties and what precisely they do. Information is also provided on how these properties are used in a theme. Some properties (i.e., alphalist-type) are only useful when used in conjunction with a list control in a theme.

An appendix is included which details the available methods for the ZonePlayerController.wsc and ZonePlayerFinder.wsc. Knowledge of these is not necessary for standard use of the script in HouseBot.

Osler

ZonePlayer Device Properties:

Script Performance / Communication

ZPScriptInterval (alphanumeric)

This property sets the frequency at which the script device queries the WSC component for information that may have arrived as a result of a call back from the ZonePlayer. When the ZonePlayer is set to play, this is also the interval at which the ZonePlayer is polled for track information to allow for a second-by-second update of track position. A value of 500 ms is within UPnP specifications and should give satisfactory performance in most situations.

ZPHHMessage (alphanumeric)

This property is populated via the use of “Magic Songs” in your media database and the handheld (HH) controller (CR100). When one of these songs is added to the queue, HouseBot will see the addition, pass the message associated with that addition to this property, and then delete the song from the queue. You can use a task to parse the messages and perform functions as needed. A message consists of the Artist information concatenated with the Title information (<Artist - Title>). In the example songs, the Album is *Media Control and the Artist is *Family Room. The title gives the precise command to execute for Family Room (i.e., +TV On, +Receiver On, +Receiver Volume Up, etc.). Note that the asterix is stripped from the message prior to passing to HouseBot (adding +TV On by artist *Family Room to the queue would pass the message Family Room - TV On to HouseBot).

To edit the sample .mp3 files to your needs, simply right-click, select summary, choose advanced, and edit the Album, Artist, Title information. **The Title must be preceded by a + for the script to recognize it as a command.** The asterix is used to ensure these special songs appear at the top of the music library when viewed in the Sonos handheld controller. I would recommend grouping functionality by Album (i.e., lighting, AV control, etc.) and grouping areas of control by Artist (i.e., pool, bedroom, etc.).

Once you have your songs edited, place the folder within you Music folder and reindex your music (via handheld controller, desktop controller, or see below).

ZPForcedResubscription (boolean)

By setting the value to 1, the script will be forced to sever communication with the ZonePlayer and then re-establish the connection at a specified time interval (see **ZPResubscriptionInterval**). This is useful in network situations where the script can lose connection with the ZonePlayer on a relatively frequent basis. By issuing a forced resubscription, you ensure that the script is executing against a connected ZonePlayer, minimizing the risk of script failure. This should only be used if you

find that there is frequent loss of communication between HouseBot and the ZonePlayer.

ZPResubscriptionInterval (alphanumeric)

The time in minutes between forced resubscriptions. If **ZPForcedResubscription** is set to false (value = 0), then this is the frequency with which the script will ping the ZonePlayer to ensure it is available on the network. If a ping fails, the script will automatically disconnect and then attempt to re-establish a connection with the ZonePlayer. A value of 10 is reasonable for ping-only function. If forced resubscriptions are being used, the interval should be tailored to your network.

Device Identification/Transport/Rendering

UDN (alphanumeric)*

This property is the unique device name (UDN) of the ZonePlayer to be controlled. **This property value must be input by the user prior to running the script.* The UDN can be found by running the UniqueDeviceName script included with the download.

ZPTransportState (alphanumeric)

This property reflects or sets the current transport state of the ZonePlayer. Allowed values are [**Play, Pause, Stop, Previous, Next**]. Allowed values can be set via the Property Manager in HouseBot.

ZPPlayMode (alphanumeric)

This property sets the current play mode of the ZonePlayer. Allowed values are [**Normal, Shuffle, Shuffle No Repeat, Repeat All**]. Allowed values can be set via the Property Manager in HouseBot.

ZPVolume (alphanumeric)

This property reflects or sets the current volume level of the ZonePlayer. Allowed values are [**0 – 100**]. Allowed values can be set via the Property Manager in HouseBot.

ZPRampToVolume (alphanumeric)

This property sets a new volume level for the ZonePlayer (either above or below the current volume) and incrementally changes the current volume to the new volume. Allowed values are [**0 – 100**]. Allowed values can be set via the Property Manager in HouseBot.

ZPRestorePreRampVolume (boolean)

When set to **1**, this property causes the ZonePlayer to return to its original volume prior to setting **ZPRampToVolume**. This would be useful in a situation where the volume is ramped down when the phone or doorbell rings and the original listening level is required to be reset.

ZPBass (alphanumeric)

This property reflects or sets the current bass level of the ZonePlayer. Allowed values are [**-10 – 10**]. Allowed values can be set via the Property Manager in HouseBot.

ZPTreble (alphanumeric)

This property reflects or sets the current treble level of the ZonePlayer. Allowed values are [**-10 – 10**]. Allowed values can be set via the Property Manager in HouseBot.

ZPMuteStatus (boolean)

This property reflects or sets the current mute state of the ZonePlayer. A value of **1** corresponds to mute on and a value of **0** corresponds to mute off.

ZPLoudnessStatus (boolean)

This property reflects or sets the current loudness state of the ZonePlayer. A value of **1** corresponds to loudness on and a value of **0** corresponds to loudness off.

Network/Group Coordination

ZPName (alphanumeric)

This property reflects the zone name of the ZonePlayer. This is pulled automatically by the script and does not require user input.

ZPIPAddress (alphanumeric)

This property reflects the IP address of the ZonePlayer. This is pulled automatically by the script and does not require user input.

ZPLinked (boolean)

If the ZonePlayer is linked as a slave, a value of **1** will be displayed. Otherwise, a value of **0** is displayed.

ZPUnlink (boolean)

If this property value is set to **1**, the ZonePlayer will unlink from all other ZonePlayers.

ZPGroupCoordinator (alphanumeric)

This property reflects the current coordinator of the ZonePlayer. If the ZonePlayer is not linked or is the coordinator of a linked group, this will simply be the zone name. If the ZonePlayer is linked as a slave, this property will reflect the name of the zone that is coordinating the ZonePlayer. This property can be used in a theme with an indicator control to reflect the link state of a ZonePlayer.

ZPLinkedZones (alphanumeric)

This property reflects the zones that are currently linked to the ZonePlayer. The field will populate dynamically as zones are linked to the ZonePlayer. The populated field is exemplified by the following: + Zone1 + Zone2 + Zone3…+ZoneN. This is meant to be used in a theme in conjunction with the ZonePlayer name, providing visual information about zones that are slaves to the ZonePlayer (i.e., a property label populated with **ZPName** and a property label adjacent to it populated with **ZPLinkedZones**).

Media: Information

ZPCurrentArtist (alphanumeric)

This property reflects the artist of the currently playing track of the ZonePlayer. This value is set automatically by the script.

ZPCurrentAlbum (alphanumeric)

This property reflects the album of the currently playing track of the ZonePlayer. This value is set automatically by the script.

ZPCurrentTrack (alphanumeric)

This property reflects the currently playing track of the ZonePlayer. This value is set automatically by the script.

ZPNextTrack (alphanumeric)

This property reflects the next track to be played. This value is automatically set by the script.

ZPTrackLength (alphanumeric)

This property reflects the total length of the currently playing track in the format [**00:00**]. This value is automatically set by the script.

ZPTrackPosition (alphanumeric)

This property reflects the current position within the currently playing track in the format [**00:00**]. This value is automatically set by the script.

ZPTrackPosLen (alphanumeric)

This property is a composite of the current position and the track length for easier use in theme construction. The format is [**00:00 / 00:00**]. This value is automatically set by the script.

ZPTrackPercent (alphanumeric)

This property reflects the percentage of the currently playing track that has been played. It is to be used with a gauge control in a theme. This value is automatically set by the script.

ZPNowPlayingPath (alphanumeric)

This property reflects the path to **album art** for the currently playing track. It is meant to be used with a dynamic image control within a theme. This value is automatically set by the script.

ZPNextPlayingPath (alphanumeric)

This property reflects the path to **album art** for the next track in the queue. It is meant to be used with a dynamic image control within a theme. This value is automatically set by the script.

ZPDisplay[1..2..3..4] (alphanumeric)

These properties are meant to be used with a property label in a theme to dynamically update a “Now Playing” screen with information on the current media. These values are automatically set by the script.

Media: Selection/Management

ZPRemoveURI (alphanumeric)

This property is used for queue management. This value is meant to be filled from within a theme via property value substitution from the property **ZPQueueSelection** (a list control selection). When populated, the script will **remove the selected track from the queue**.

ZPAddURI (alphanumeric)

This property is used for queue management. This value is meant to be filled from within a theme via property value substitution from the property **ZPBrowseSelection** (a list control selection). When populated, the script will **add the selected track to the end of the queue**.

ZPPlayNowURI (alphanumeric)

This property is used for queue management. This value is meant to be filled from within a theme via property value substitution from the property **ZPBrowseSelection** or **ZPQueueSelection** (a list control selection). When populated, the script will **add the selected track to the end of the queue and play it**.

ZPPlayNextURI (alphanumeric)

This property is used for queue management. This value is meant to be filled from within a theme via property value substitution from the property **ZPBrowseSelection** or **ZPQueueSelection** (a list control selection). When populated, the script will **add the selected track to the queue to a position after the currently playing track**.

ZPClearPlayURI (alphanumeric)

This property is used for queue management. This value is meant to be filled from within a theme via property value substitution from the property **ZPBrowseSelection** (a list control selection). When populated, the script will **clear the queue and then add the selected track to the queue**.

ZPReindexMusic (boolean)

When set to true, this property value will cause the music library to be reindexed. It must be used in conjunction with **ZPIndexType**.

ZPIndexType (alphanumeric)

Allowed values are ALBUM or ALBUMARTIST.

Media: Queue

ZPQueue (alphalist)

This property provides a list of the queue. Used in a theme with a list control, the items in the queue can be displayed. This value is automatically set by the script.

ZPQueuePosition (alphanumeric)

This property reflects the position of the current track in the queue. This value is automatically set by the script.

ZPQueueMoveUp (alphanumeric)

Not implemented.

ZPQueueMoveDown (alphanumeric)

Not implemented.

ZPClearQueue (boolean)

When this property is set to a value of **1**, the queue will be cleared.

ZPQueueSize (alphanumeric)

This property reflects the size of the queue. This value is automatically set by the script.

ZPQueuePosSize (alphanumeric)

This property is a composite of the queue position and the queue size for easier use in theme construction. The format is [**0 / 0**]. This value is automatically set by the script.

ZPQueueSelection (alphanumeric)

This property reflects the currently selected item when **ZPQueue** is used with a list control in a theme. See **ZPRemoveURI**, **ZPPlayNowURI**, **ZPPlayNextURI**.

Media: Library

ZPBrowse (alphanumeric)

This property provides control for moving through the music library. Allowed values are [**Forward, Back, Up, Down, Hold**]. Allowed values can be set via the Property Manager in HouseBot.

ZPBrowseIndex (alphanumeric)

This property keeps track of the level to which the music menu has been “drilled” into. This value is automatically set by the script.

ZPBrowseList (alphalist)

This property provides a list of available media in the music library. Used in a theme with a list control, the items in the media library can be displayed. This value is automatically set by the script. Navigation of the list is done via the property **ZPBrowse**.

ZPBrowseTotalItems (alphanumeric)

This property reflects the total number of items in the selected sub-heading of the media library (Album, Artist, Track, etc.).

ZPBrowseReturn (alphanumeric)

This is the total number of items to be returned in **ZPBrowseList** when a browse is executed. This allows the list size to be tailored to the theme to prevent the windows scrollbars from appearing in the list control.

ZPBrowseSelection (alphanumeric)

This property reflects the currently selected item when **ZPBrowseList** is used with a list control in a theme. See **ZPAddURI**, **ZPPlayNowURI**, **ZPPlayNextURI**, **ZPClearPlayURI**.

ZPMusicMenuTitle (alphanumeric)

This property displays the current position within the music menu. This value will change as you “drill down” to your music selection. It is meant to be used with a property label in a theme in conjunction with a list control reflecting the **ZPBrowseList** property. This value is automatically set by the script.

ZPBrowsePercent (alphanumeric)

This provides the position of **ZPBrowseList** in relation to **ZPTotalItems** as a percent. It is meant to be used with a gauge control and **ZPBrowse** (Up and Down) to allow for a custom scrollbar to be created in a theme.

ZPSearchString (alphanumeric)

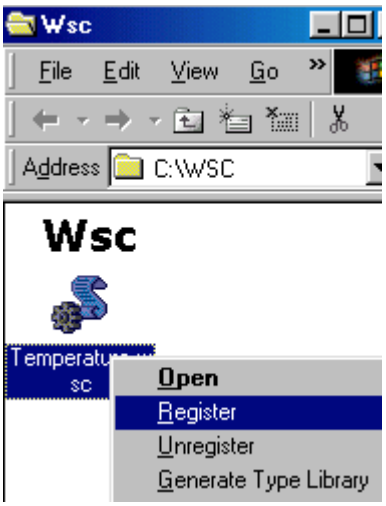
This provides the ability to use the power scroll function built into the ZonePlayer. It is only available when searching by Album, Artist, or Track. Acceptable values are #, A, B, C, ….W, X, Y, Z.

ZPSearchType (alphanumeric)

This allows for the quick movement between search types (i.e., perform a search and selection by Album and then immediately move to a search by Track without having to back all the way out of the current search). Acceptable values are A:Album, A:Artist, A:Tracks, A:Playlists (imported playlists), A:Genre, A:Composer, AI: (audio in), S: (folder view), SQ: (Sonos playlists), R: (radio in its pre-Radio Time format), Q:0 (queue).

Appendix.

ZonePlayerController Class



REGISTERING THE SCRIPT COMPONENT
The component and its type library can be placed anywhere, but I would recommend creating a root directory called WSC and placing all windows script components there. To use the component, it must first be registered. The component can be registered as if it were a DLL using the command prompt and regsrv32. The easier method is to right-click the component and choose Register from the pop-up menu (see figure at left).

Included with the component is a type library with the uninteresting name "scriptlet TLB file". If you use a program such as Vbsedit or similar to write scripts, the type library is necessary to allow you to visualize the available methods of the component as you write your scripts. If the type library was not included in the download, simply right-click the component and choose Generate Type Library from the pop-up menu.

CREATING THE COM OBJECT
Before the methods of the object can be called, it must first be created and assigned to an object variable. The following snippet shows how to create the object.

```
1       Dim FamilyRoom          'As ZonePlayer.Controller
2
3       Set FamilyRoom = CreateObject ("ZonePlayer.Controller)"
```

CALL METHODS

1. Connect(ByVal strUniqueDeviceName)

Syntax: ReturnCode = Object.Connect("UniqueDeviceName")

Parameter1: strUniqueDeviceName specifies the UniqueDeviceName of the ZonePlayer in the format "uuid:RINCON1234567890…"

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.Connect("uuid:RINCON1234567890…")

MsgBox(FamilyRoom.Connect("uuid:RINCON1234567890…"))
```

Note: This method must be called before any other method in the object can be used.

2. Disconnect()

Syntax: Object.Disconnect

ReturnCode: None

Sample:

```
FamilyRoom.Disconnect
```

Note: This method must be called before the object is destroyed.

3. SetTransportState(ByVal strNewState)

Syntax: ReturnCode = Object.SetTransportState("NewState")

Parameter1: strNewState specifies the transport state to change to. Allowed states are "Play", "Pause", "Stop", "Next", "Previous".

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetTransportState("Play")

MsgBox(FamilyRoom.SetTransportState("Pause"))
```

4. GetTransportState()

Syntax: ReturnCode = Object.GetTransportState

ReturnCode: If successful, the current transport state will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentTransportState

CurrentTransportState = FamilyRoom.GetTransportState
```

5. SetVolumeLevel(ByVal strChannel, ByVal intNewLevel)

Syntax: ReturnCode = Object.SetVolumeLevel("Channel", NewLevel)

Parameter1: strChannel specifies the channel to control. Allowed values are "Master", "LF", "RF".

Parameter2: intNewLevel specifies the new volume level. This should be an integer from 0 to 100.

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetVolumeLevel("Master", 50)

MsgBox(FamilyRoom.SetVolumeLevel("Left", 25))
```

6. GetVolumeLevel()

Syntax: ReturnCode = Object.GetVolumeLevel

ReturnCode: If successful, the current volume level will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentVolumeLevel

CurrentVolumeLevel = FamilyRoom.GetVolumeLevel
```

7. SetBassLevel(ByVal intNewLevel)

Syntax: ReturnCode = Object.SetBassLevel(“NewLevel”)

Parameter1: intNewLevel specifies the new bass level. This should be an integer from -10 to 10.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetBassLevel(5)

MsgBox(FamilyRoom.SetBassLevel(7))
```

8. GetBassLevel()

Syntax: ReturnCode = Object.GetBassLevel

ReturnCode: If successful, the current bass level will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentBassLevel

CurrentBassLevel = FamilyRoom.GetBassLevel
```

9. SetTrebleLevel(ByVal intNewLevel)

Syntax: ReturnCode = Object.SetTrebleLevel(“NewLevel”)

Parameter1: intNewLevel specifies the new bass level. This should be an integer from -10 to 10.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.TrebleLevel(5)

MsgBox(FamilyRoom.SetTrebleLevel(7))
```

10. GetTrebleLevel()

Syntax: ReturnCode = Object.GetTrebleLevel

ReturnCode: If successful, the current bass level will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentTrebleLevel

CurrentTrebleLevel = FamilyRoom.GetTrebleLevel
```

11. SetMuteState(ByVal strChannel, ByVal blnNewState)

Syntax: ReturnCode = Object.SetMuteState(“Channel”, NewState)

Parameter1: strChannel specifies the channel to control. Allowed values are “Master”, “LF”, “RF”.

Parameter2: blnNewState specifies the new state. True corresponds to muted and False corresponds to unmated.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetMuteState(“Master”, True)

MsgBox(FamilyRoom.SetMuteState(“Left”, False))
```

12. GetMuteState()

Syntax: ReturnCode = Object.GetMuteState

ReturnCode: If successful, the current mute state will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentMuteState

CurrentMuteState = FamilyRoom.GetMuteState
```

13. SetLoudnessState(ByVal strChannel, ByVal blnNewState)

Syntax: ReturnCode = Object.SetLoudnessState(“Channel”, NewState)

Parameter1: strChannel specifies the channel to control. Allowed values are “Master”, “LF”, “RF”.

Parameter2: blnNewState specifies the new state. True corresponds to loudness on and False corresponds to loudness off.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetLoudnessState(“Master”, True)

MsgBox(FamilyRoom.SetLoudnessState(“Left”, False))
```

14. GetLoudnessState()

Syntax: ReturnCode = Object.GetLoudnessState

ReturnCode: If successful, the current loudness state will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentLoudnessState

CurrentLoudnessState = FamilyRoom.GetLoudnessState
```

15. SetLEDState(ByVal strNewState)

Syntax: ReturnCode = Object.SetLEDState(“NewState”)

Parameter1: strNewState specifies the state of the white LED on the ZonePlayer. Allowed values are “On” and “Off”.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetLEDState("On")

MsgBox(FamilyRoom.SetLEDState("Off"))
```

16. GetLEDState()

Syntax: ReturnCode = Object.GetLEDState

ReturnCode: If successful, the current LED state will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentLEDState

CurrentLEDState = FamilyRoom.GetLEDState
```

17. SetZoneName(ByVal strNewName, ByVal strNewIcon)

Syntax: ReturnCode = Object.SetZoneName("NewName", "NewIcon")

Parameter1: strNewState specifies the zone name of the ZonePlayer. Any reasonable name can be used.

Parameter2: strNewIcon specifies the zone icon to be displayed in the Desktop Controller. It must have the format x-rincon-roomicon:<icon>. All allowed values of <icon> are currently not known; however, "family" "living', "dining", "bedroom", "patio", "pool", "media", "garden", "bathroom", "hallway", "den", "garage", "office", "foyer", "library", and "kitchen" are known values. If a non-empty string is passed that doesn't correspond to a known icon, the default icon is a table lamp. Alternatively, an empty string can be passed which the ZonePlayer will interpret as a name change only (icon left as previously set).

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetZoneName("My ZonePlayer", "")

MsgBox(FamilyRoom.SetZoneName("My ZonePlayer", ""))
```

18. GetZoneName()

Syntax: ReturnCode = Object.GetZoneName

ReturnCode: If successful, the current zone name will be returned. Otherwise an error will be returned.

Sample:

```
Dim CurrentZoneName

CurrentZoneName = FamilyRoom.GetZoneName
```

19. GetSerialNumber()

Syntax: ReturnCode = Object.GetSerialNumber

ReturnCode: If successful, the ZonePlayer serial number will be returned. Otherwise an error will be returned.

Sample:

```
Dim SerialNumber

SerialNumber = FamilyRoom.GetSerialNumber
```

20. GetSoftwareVersion()

Syntax: ReturnCode = Object.GetSoftwareVersion

ReturnCode: If successful, the ZonePlayer software version will be returned. Otherwise an error will be returned.

Sample:

```
Dim SoftwareVersion

SoftwareVersion = FamilyRoom.GetSoftwareVersion
```

21. GetDisplaySoftwareVersion()

Syntax: ReturnCode = Object.GetDisplaySoftwareVersion

ReturnCode: If successful, the software version of the Desktop Controller will be returned. Otherwise an error will be returned.

Sample:

```
Dim DisplaySoftwareVersion

DisplaySoftwareVersion = FamilyRoom.GetDisplaySoftwareVersion
```

22. GetHardwareVersion()

Syntax: ReturnCode = Object.GetHardwareVersion

ReturnCode: If successful, the ZonePlayer hardware version will be returned. Otherwise an error will be returned.

Sample:

```
Dim HardwareVersion

HardwareVersion = FamilyRoom.GetHardwareVersion
```

23. GetIPAddress()

Syntax: ReturnCode = Object.GetIPAddress

ReturnCode: If successful, the ZonePlayer IP address will be returned. Otherwise an error will be returned.

Sample:

```
Dim IPAddress

IPAddress = FamilyRoom.GetIPAddress
```

24. GetMACAddress()

Syntax: ReturnCode = Object.GetMACAddress

ReturnCode: If successful, the ZonePlayer MAC address will be returned. Otherwise an error will be returned.

Sample:

```
Dim MACAddress

MACAddress = FamilyRoom.GetMACAddress
```

25. GetCopyrightInfo()

Syntax: ReturnCode = Object.GetCopyrightInfo

ReturnCode: If successful, the copyright info will be returned. Otherwise an error will be returned.

Sample:

```
Dim CopyrightInfo

CopyrightInfo = FamilyRoom.CopyrightInfo
```

26. GetHouseholdID()

Syntax: ReturnCode = Object.GetHouseholdID

ReturnCode: If successful, the household ID of sonos-net will be returned. Otherwise an error will be returned.

Sample:

```
Dim HouseholdID

HouseholdID = FamilyRoom.GetHouseholdID
```

27. SetLineInLevel(ByVal intLeftLevel, ByVal intRightLevel)

Syntax: ReturnCode = Object.SetLineInLevel(LeftLevel, RightLevel)

Parameter1: intLeftLevel specifies the left line-in level. This should be an integer from 0 to 15.

Parameter2: intRightLevel specifies the left line-in level. This should be an integer from 0 to 15.

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SetLineInLevel(5, 5)

MsgBox(FamilyRoom.SetLineInLevel(2, 8))
```

26. GetLeftLineInLevel() [GetRightLineInLevel()]

Syntax: ReturnCode = Object.GetLeftLineInLevel

ReturnCode: If successful, the left line-in level will be returned. Otherwise an error will be returned.

Sample:

```
Dim LeftLineIn

LeftLineIn = FamilyRoom.GetLeftLineInLevel
```

27. GetQueue()

Syntax: ReturnCode = Object.GetQueue

ReturnCode: If successful, the current queue will be returned as a two-dimensional array. The array (x,y) can be viewed as a table where the value of x represents an entry or row of the table and y represents the columns of the table. The lower bounds of both x and y is zero. The upper bounds of x is the total number of tracks in the queue minus 1 and the upper bounds of y is 5. The data returned is as follows:

(x, 0) = Artist
(x, 1) = Album
(x, 2) = Track
(x, 3) = Track URI
(x, 4) = AlbumArt URL
(x, 5) = ObjectID

Sample:

```
Dim Queue
Dim Artist, Track

'Info for 10th track in queue
Queue = FamilyRoom.GetQueue
Artist = Queue(9, 0)
Track = Queue(9, 2)
```

28. GetCurrentTrackInfo()

Syntax: ReturnCode = Object.CurrentTrackInfo

ReturnCode: If successful, the currently playing track info will be returned as a one-dimensional array. The lower bounds of the array is 0 and the upper bounds of the array is 0. The data returned is as follows

(0) = Artist
(1) = Album
(2) = Track (Or Linked, Streaming, Internet Radio)
(3) = Track URI
(4) = Album Art URI
(5) = Track Position
(6) = Track Duration
(7) = Queue Position
(8) = Percent Played

Sample:

```
Dim TrackInfo
Dim Title

TrackInfo = FamilyRoom.GetCurrentTrackInfo
Title = TrackInfo(2)
```

29. ClearQueue()

Syntax: ReturnCode = Object.ClearQueue

ReturnCode: If successful, the current queue will be cleared and “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.ClearQueue

MsgBox(FamilyRoom.ClearQueue)
```

30. RemoveTrackFromQueue(ByVal strObjectID)

Syntax: ReturnCode = Object.RemoveTrackFromQueue("ObjectID")

Parameter1: strObjectID specifies the ID of the song to be removed. This ID can be found via the function GetQueue.

Return Code: If successful, "OK" will be returned. Otherwise an error will be returned.

```
FamilyRoom.RemoveTrackFromQueue("Q:0/1")
```

31. AddSongToQueue(ByVal strURI, ByVal strMetaData)

Syntax: ReturnCode = Object.AddSongToQueue("URI", "MetaData")

Parameter1: strURI specifies the URI of the song to be added.

Parameter2: strMetaData specifies the xml metadata of the track to be added. This parameter is optional, but an empty string ("") must be passed if no metadata is passed.

Return Code: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
Dim URI
URI = "x-file-
cifs://MEDIASERVER/iTunes%20Music/Marc%20Seales,%20composer.
%20New%20Stories.%20Erni/Speakin'%20Out/01%20_Highway%20Blu
es_.m4a"

FamilyRoom.AddSongToQueue(URI, "")
```

32. PlaySongNow(ByVal strURI, ByVal strMetaData) [see also Link()]

Syntax: ReturnCode = Object.AddSongToQueue("URI", "MetaData")

Parameter1: strURI specifies the URI of the song to be added. This is also the method by which zones are linked together. If the string:

"x-rincon: <UniqueDeviceName>"

(UniqueDeviceName has the form "RINCON_000E12345…" as when using the Connect method presented above minus "uuid:") is passed in the strURI parameter, the current ZonePlayer will be linked to the ZonePlayer which the unique device name belongs to.

Parameter2: strMetaData specifies the xml metadata of the track to be added. This parameter is optional, but an empty string ("") must be passed if no metadata is passed.

Return Code: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
Dim URI
URI = “x-file-cifs://MEDIASERVER/iTunes%20Music/Marc%20Seales,%20composer.%20New%20Stories.%20Erni/Speakin'%20Out/01%20_Highway%20Blues_.m4a”

FamilyRoom.PlaySongNow(URI, “”)
```

33. PlayMode_RepeatAll()

Syntax: ReturnCode = Object.PlayMode_RepeatAll

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.PlayMode_RepeatAll

MsgBox(FamilyRoom.PlayMode_RepeatAll)
```

34. PlayMode_Shuffle()

Syntax: ReturnCode = Object.PlayMode_Shuffle

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.PlayMode_Shuffle

MsgBox(FamilyRoom.PlayMode_Shuffle)
```

35. PlayMode_ShuffleNoRepeat()

Syntax: ReturnCode = Object.PlayMode_ShuffleNoRepeat

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.PlayMode_ShuffleNoRepeat

MsgBox(FamilyRoom.PlayMode_ShuffleNoRepeat)
```

36. PlayMode_Normal()

Syntax: ReturnCode = Object.PlayMode_Normal

ReturnCode: If successful, “OK” will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.PlayMode_Normal

MsgBox(FamilyRoom.PlayMode_Normal)
```

37. ReindexMusic(ByVal strAlbumArtistDisplayOption)

Syntax: ReturnCode = Object.ReindexMusic("DisplayOption")

Parameter1: strAlbumArtistDisplayOption function is unknown. An empty string ("") can be passed.

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.ReindexMusic("")

MsgBox(FamilyRoom.ReindexMusic(""))
```

38. RefreshShareList()

Syntax: ReturnCode = Object.RefreshShareList

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.RefreshShareList

MsgBox(FamilyRoom.RefreshShareList)
```

39. RefreshRadioStationList()

Syntax: ReturnCode = Object.RefreshRadioStationList

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.RefreshRadioStationList

MsgBox(FamilyRoom.RefreshRadioStationList)
```

40. GetLastReindexDate()

Syntax: ReturnCode = Object.GetLastReindexDate

ReturnCode: If successful, the date available music was last indexed will be returned.

Sample:

```
Dim Date

Date = FamilyRoom.GetLastReindexDate
```

41. Unlink()

Syntax: ReturnCode = Object.Unlink

ReturnCode: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.Unlink

MsgBox(FamilyRoom.Unlink)
```

Note: This action unlinks the ZonePlayer from other ZonePlayers.

42. Link(ByVal strUDN)

Syntax: ReturnCode = Object.Link("UDN")

Parameter1: strUDN specifies the UDN of the ZonePlayer to link to in the format:

"x-rincon:<UniqueDeviceName>"

(UniqueDeviceName has the form "RINCON_000E12345…" as when using the Connect method presented above but minus "uuid:")

Return Code: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.Link("x-rincon:RINCON_0000E1234567")
```

43. AVTransportCallback()

Syntax: ReturnCode = Object.AVTransportCallback

Return Code: If successful, the current transport service info will be returned as a one-dimensional array. The lower bounds of the array is 0 and the upper bounds of the array is 8. The data returned is as follows

(0) = Transport State
(1) = Queue Position
(2) = Track Length
(3) = Queue Size
(4) = Current Artist
(5) = Current Track
(6) = Current Album
(7) = Next Artist
(8) = Next Track
(9) = Next Album
(10) = Current URI
(11) = Current Album Art URL
(12) = Next Album Art URL

Sample:

```
see below
```

44. RenderingCallback()

Syntax: ReturnCode = Object.VolumeCallback

Return Code: If successful, the current rendering service info will be returned as a one-dimensional array. The lower bounds of the array is 0 and the upper bounds of the array is 9. The data returned is as follows

(0) = Master Volume
(1) = Left Volume
(2) = Right Volume
(3) = Master Mute
(4) = Left Mute
(5) = Right Mute

(6) = Treble
(7) = Bass
(8) = Loudness
(9) = Output Fixed

Sample:

```
see below
```

45. PropertiesCallback()

Syntax: ReturnCode = Object.VolumeCallback

Return Code: If successful, the current properties service info will be returned as a one-dimensional array. The lower bounds of the array is 0 and the upper bounds of the array is 3. The data returned is as follows

(0) = Zone Name
(1) = Zone Icon
(2) = Zone Invisible
(3) = Settings Replication State

46. CheckCallbackStatus(ByVal intIndex)

Syntax: ReturnCode = Object.CheckCallbackStatus

Return Code: This returns a Boolean value reflecting if an update from the Zone Player has occurred. A value of True indicates an update has occurred. The returned Boolean is a 1-dimensional array with upper-bound of 2. Array position 0 returns True if a transport callback has been received, array position 1 returns True if a rendering callback has been received, and array position 2 returns True if a property callback has been received.

Sample:

```
Dim TransportCallback

TransportCallback = FamilyRoom.CheckCallbackStatus(0)
```

47. ResetCallbackStatus(ByVal intIndex)

Syntax: ReturnCode = Object.ResetCallbackStatus

Return Code: This function resets the value of CallbackStatus within the component to False. See item 46 above. This should be used after a status update from the ZonePlayer has occurred to reset the “flag” so that future updates can be discerned.

Sample:

```
FamilyRoom.ResetCallbackStatus(0)
```

Sample Use of CallbackStatus Functions:

```
Do
        Wscript.Sleep(500)
        TransportCallback
Loop

Private Sub TransportCallback()

        If FamilyRoom.CheckCallbackStatus(0) = True Then

                Dim TransportData
                Dim TranpsportState

                TransportData = FamilyRoom.AVTransportCallback
                TransportState = TransportData(0)

                FamilyRoom.ResetCallbackStatus(0)
        End If

End Sub
```

Note: These functions must be used in a continually running script or executable to function correctly.

48. RampToVolume(ByVal strChannel, ByVal intDesiredVolume)

Syntax: ReturnCode = Object.RampToVolume("Channel", Volume)

Return Code: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.RampToVolume("Master", 30)
```

49. SeekTrack(ByVal intTrackNumber)

Syntax: ReturnCode = Object.SeekTrack(TrackNumber)

Return Code: If successful, "OK" will be returned. Otherwise an error will be returned.

Sample:

```
FamilyRoom.SeekTrack(3)
```

50. ChangeTrackPosition(ByVal intTrackNumber, ByVal intNewPosition)

51. Browse(ByVal strBrowseContainer, ByVal strRequestedCount, ByVal strStartingIndex)

52. RadioStationCatalog()

53. GetGroupCoordinator()

54. GetMessageBuffer()

55. GetObjectMetadata(ByVal strObjectID)

ZonePlayerFinder Class

1. **FindZonePlayers()**
2. **GetZoneCount()**
3. **GetZoneInfo()**
4. **GetUDNInfo()**
5. **GetZoneName(ByVal intZoneNumber)**
6. **GetUDNByZoneName(ByVal strZoneName)**
7. **GetZoneUDN(ByVal intZoneNumber)**