Yamaha AV Receiver HTTP plugin

Document Version 2 - February 2014
By A-Lurker, XML parser by Futzle

Motivation:

There already exists a good Yamaha Receiver Plugin - so why make another one?

The original plugin in Vera uses a direct socket connection to the receiver. If the receiver's mains power is removed completely, as opposed to just Standby, then when the power is subsequently restored, you find that the connection is not re-established.

At this point the only way to re-establish contact with the receiver is to restart the Luup engine. As I understand it, this is a restriction imposed by Vera's current firmware, when using the functions in the luup io module: as of Feb 2014 - version: 1.5.622

The Yamaha receiver can also be controlled using an HTTP connection - ie with a web browser. This plugin emulates that control. This has the advantage, that after a power outage to the receiver, a connection can always be re-established to the receiver.

Set up using the Advanced Tab:

The plugin is primarily designed to configure the device initially – ready to play. The preference is to use the Advance tab in the scene editor to set up the commands, rather than providing any dashboard controls. If you want to select what's actually played, try out the DLNA plugin.

Note that all variables are case sensitive. For example 'TUNER' (used by the 'Input' parameter) is not the same as 'Tuner' (used by the 'Source' parameter).

The easiest way to see what inputs are available is to look at what's used in the receiver's web page. There is a link to that in the Device's Control Tab.

Typical values often used:

```
Zone:   'Main_Zone', 'Zone_2', 'Zone_3', 'Zone_4'   or you can use  1, 2, 3, 4
instead

Input:   'HDMIx', 'AVx', 'AUDIOx', 'V-AUX', 'TUNER', 'USB', 'Pandora',
'AirPlay', 'SERVER', etc

Source:   'SERVER', 'USB', 'NET RADIO', 'iPod_USB', 'Pandora' , 'Tuner', etc
```

More details can be found further below and in this spreadsheet. To view the spreadsheet, macros do not need to be allowed or its links updated:

```
calaos.fr/download/Yamaha_Spec_UK.../V671_3071_FuncTree_1.10.xls
```

Useage:
Variables you can examine, noting that all variables names are case sensitive.

The 'Connected' variable indicates if the unit is powered on and contactable via the LAN. It can take up to 30 seconds before the variable indicates the correct state. If you want to ensure, if the unit is connected, then check the 'Connected' variable first.

Despite the 30 second delay above, any command executed via the plugin, will update the status almost immediately.

Depending on your actual receiver model, some commands will work and some won't - if in doubt, just try them out after doing your own research. Also look at what's used/available in the receiver's web page.

Debug:
If you can't open the receiver's web page in a browser on your network, then this plugin is not going to work either.

Any invalid command will show up in the log file - search on the string:

```
'YamahaRxHTTP debug: Error - this command is invalid:'
```

Did I mention that alphabetic case is important?

There is a hidden variable called 'VOLFIXVAR ', that is used to mark the volume level as 'Variable' or 'Fixed'. If the latter is in force, then the volume commands become disabled.

Variables:
You can retrieve the following variables (with examples):

```
PluginVersion:   0.51
ModelName:   RX-V673
ZoneCount:   1 to 4
Connected:   0 or 1
LinkToDeviceWebPage:   Receiver's web page URL
```

And for each Zone, where x = zone number: 1 to 4

```
Zone_x_Power:     On/Standby
Zone_x_Input:    AUDIO1, etc
Zone_x_Volume:   -80.5 to 16.5
Zone_x_Mute:     On/Off
```

These statuses cannot (currently) be examined or retrieved:

```
CurrentScene number
```

```
PlayControl ie stopped, paused, etc
PartyMode state
```

Zone parameter:

Many of the calls use the Zone parameter. The amount of zones available depends on the receiver. There can be up to four of them - they can be named or numbered.

```
Zone:   1, 2, 3, 4 or 'Main_Zone', 'Zone_2', 'Zone_3', 'Zone_4'
```

Commands and their Parameters:

SetPower

```
Zone:
Power:  'On', 'Standby'
```

SetScene

```
Zone:
SceneNumber: 1 to 12
```

SetInput

```
Zone:
Input:  'HDMIx', 'AVx', 'AUDIOx', 'V-AUX', 'TUNER', 'USB', 'Pandora', 'AirPlay',
where x is an integer. There are probably others - just try them. Case matters.
```

SetVolume

```
Note - this functionality can be disabled by the 'VOLFIXVAR' variable

Zone:
Volume:  the range is -80.5 to 16.5, step size is 0.5
```

SetVolumeUpDown

```
Note - this functionality can be disabled by the 'VOLFIXVAR' variable

Zone:
Step:  increments of 0.5, 1, 2 and 5 dB are allowed. Anything else results in a
-0.5 dB step
```

SetMute

```
Zone:
Mute:  'On', 'Off', 'Att -40 dB', 'Att -20 dB', 'On/Off' (the latter being a
toggle)
```

PlayControl

```
Source:  'SERVER', 'USB', 'NET RADIO', 'iPod_USB', 'Pandora' and probably others
- just try them
Mode:  'Stop', 'Pause', 'Play', 'Skip Rev', 'Skip Fwd'

Note -'Pause' and 'Play' do not work on some models. This a Yamaha firmware
problem.
```

SelectPreset

```
Source:  'SERVER', 'USB', 'NET RADIO', 'iPod_USB', 'Pandora' and probably others
- just try them
Preset:  numbered from 1 to 40
```

SetPartyMode

```
LetsParty: 'On', 'Off'
```

SendRemoteCode

```
RemoteCode:  an 8 character hex code eg Zone 2 On/Off toggle = 7A85453A
```

Lua calls:

You can also use Lua code to arrive at the same outcomes. Here is a sample program that will run in the Lua test window in UI5:

```
-- the id of your unit
local YAMAHA_ID = your_unit_id_goes_here

-- is the receiver connected OK?
local connected = luup.variable_get("urn:a-lurker-com:serviceId:YamahaRxHTTP1",
"Connected", YAMAHA_ID)
luup.log('Connected = '..connected)

if (connected == '1') then
    luup.call_action("urn:a-lurker-com:serviceId:YamahaRxHTTP1", "SetScene",
{Zone='1', Scene='1'}, YAMAHA_ID)
end

return true
```

URLs

You can also use URLs - they must be properly escaped. Example - turn Main Zone Mute on:

```
http://VERAs_IP_Address_Here:3480/data_request?id=lu_action&serviceId=urn:a-
lurker-
com:serviceId:YamahaRxHTTP1&action=SetMute&Zone=1&Mute=On&DeviceNum=Your_Device_
Number_Here
```

All the parameters are strings, however these functions when called from other Lua code, can also accept numbers:

```
SceneNumber, Zone, Volume and VolumeUpDown
```