Harmony Controller instructions, for version 3.3 and up

Major changes with version 3.3

Version 3.0 makes use of WebSockets to communicate with your Hub on the Local LAN. No Internet access is required. By default, the plug-in will keep the WebSocket connection open at all times allowing for fast updates on the Hub status. When the activity is changed via the Harmony remote or phone app, the plug-in will now pick up on the change quickly.

The details needed to configure the Activities, Devices and Commands are stored in variables making those pages appear instantly and reducing the times the configuration needs to be pulled from the Hub.

The new ChangeChannel command makes it possible to change the channel simple for devices supporting it (e.g. TV, Radio, Sonos).

It is now also possible to use the Activity and Device names as defined on your Hub instead of the numeric IDs to start an activity or send a command.

V3.3 supports the Hub Automation interface that is used to control Philips Hue attached lights for example.

There are several other smaller enhancements and fixes.

Install from the App Market

This plug in is available on the MiOS App market as **Harmony Hub Control**. Use the standard Vera Install Apps to install.

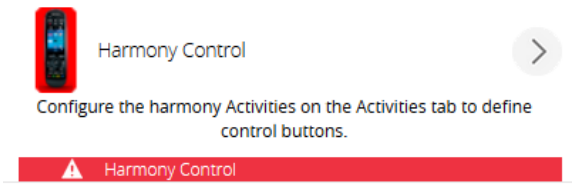
In case you have multiple Hubs you have to install multiple instances of this plug-in. Simply go to My Apps. Choose the Harmony and click Create another, and put the IP address of the second Hub in the IP address as explained below.

Install on openLuup

As of version 2.7 this plugin also work on openLuup. To install do not copy the files from your Vera, but install from the ALTUI App Store. To create a second instance of the plugin, you must create the device manually.

You can also download them from GitHub https://github.com/reneboer/vera-Harmony-Hub for a manual install.

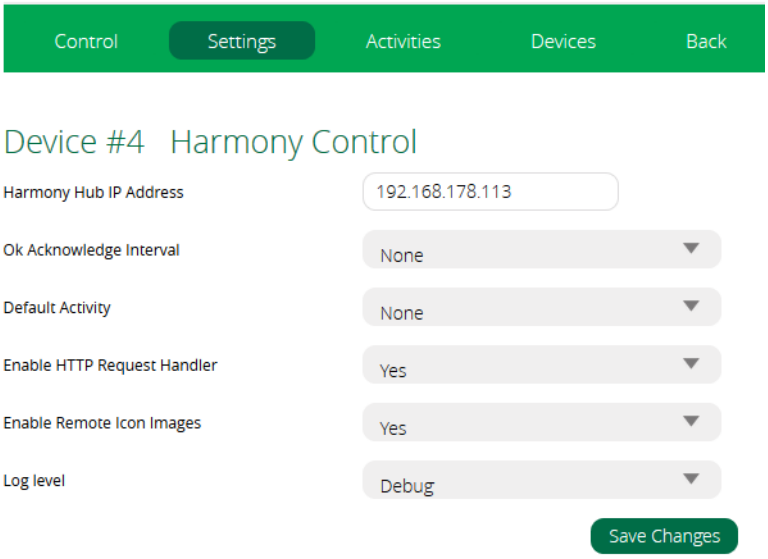
The device will show with a red icon when first installed, but not yet configured:



Configure the Harmony Hub Control Plug-in

First complete the settings page. Do not use the Advanced tab to enter any data, always use the Settings tab.

Enter the `Harmony Hub IP address`. You should configure your LAN router to always assign that same IP address to your Harmony Hub.



When `Ok acknowledge interval` is set, the icon will show with a green background after a command is completed for that number of seconds.

After you have defined activities, you can configure one of those as a Default Activity that will be selected when SetTarget = 1.

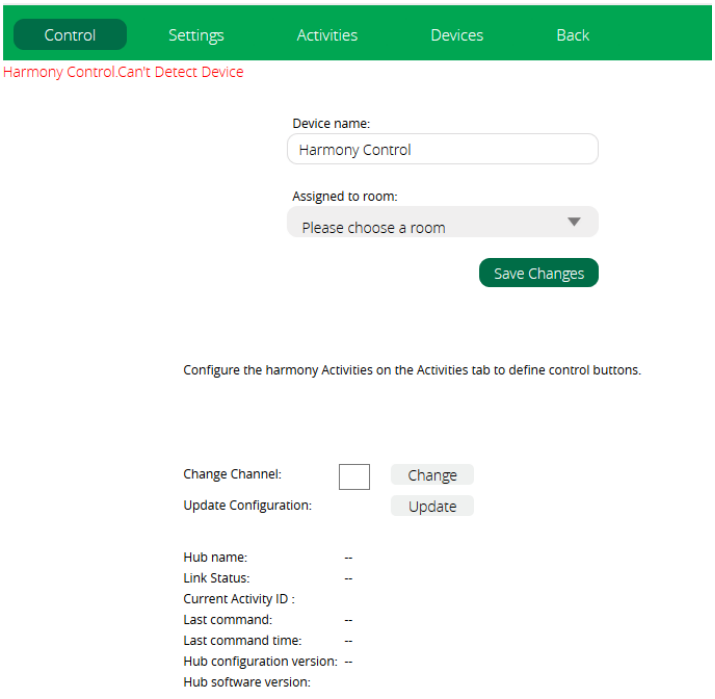
Set `Enable HTTP Request Handler` to Yes to query the harmony hub via the Vera. This is especially useful to find device commands or control via a web browser or so. See the last page for examples.

When `Enable Remote Icon Images` is Yes the Icon images will be pulled from a remote web server (github) rather than from you Vera. Benefits are that you will also see the Icons when accessing your Vera remotely. Drawback you need an internet connection, even with local use.

You can set your own preferred log level when you need to debug some odd behavior or just want to see how things flow in more detail.

When saving the settings, the Vera will do a LUUP reload to make sure all settings are properly stored and processed.

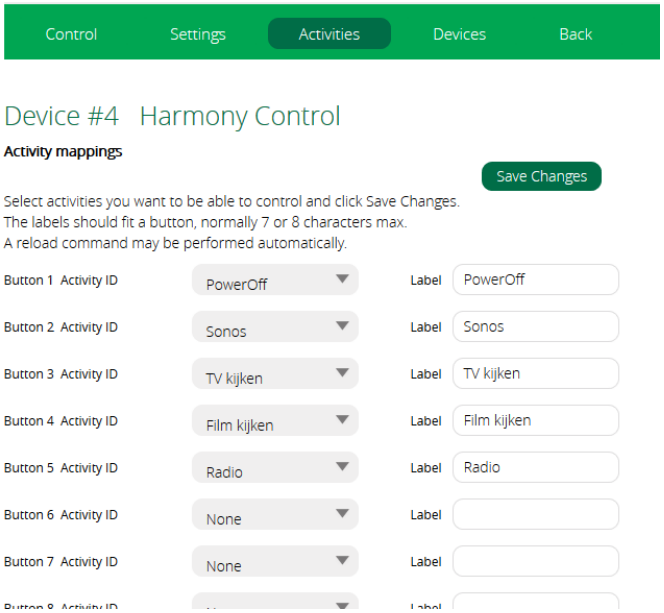
On the Control tab you should now see the message it is time to map the activities to buttons.



Set up the Harmony Activities

After saving the settings you can start to configure the activities that you want as buttons on the Vera UI.

Open the Activities Tab. You can specify up to 25 buttons as you selected in your settings. The buttons will map to one of the activities you configured on your Hub.

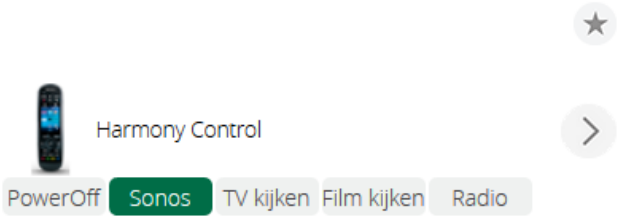


You can enter your own description in the Label input. If you leave it blank it will use the description from your Hub as shown in the dropdown.

Once you selected the activities and specified the labels for the buttons, click Save Changes.

After a few seconds you will see the message; Changes to the buttons made. Your Vera will reload, so wait a minute and refresh your browser page!

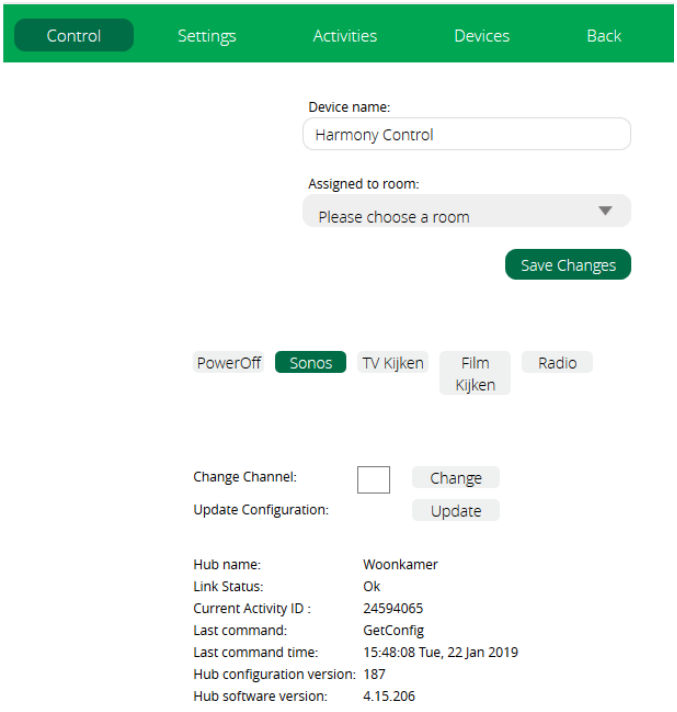
The plug-in will look something like below.



The control panel will look like this. Also showing the defined buttons.

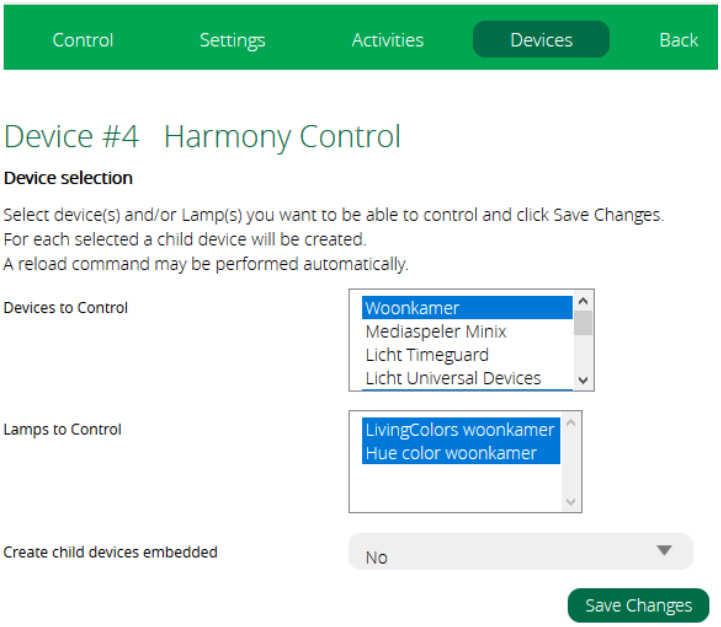
From the control panel you can send a ChangeChannel command. Enter the desired channel number in the input and click the Change button.

There could be a situation the plug-in may not have the latest Hub configuration, click Update if in doubt.



Setup Harmony Devices

Select the Devices Tab to see the device details from your Harmony Hub.



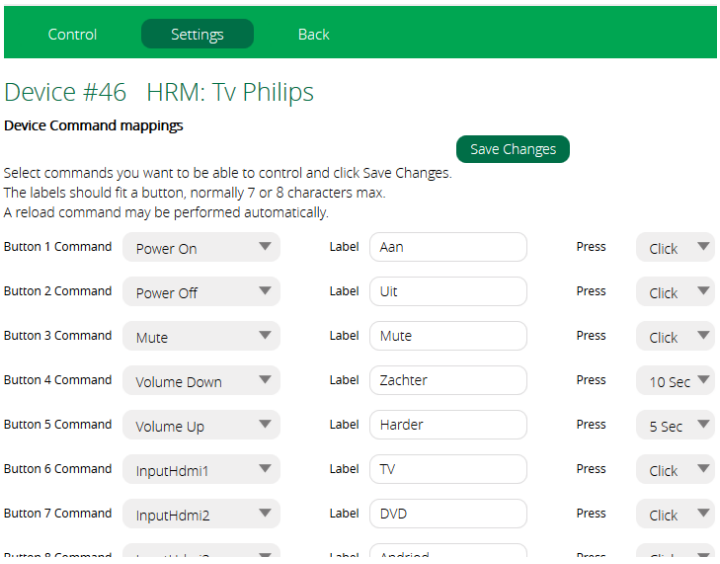
In the dropdown you can select one, several, or all your Harmony devices. For each selected device, a child device will be created. To select multiple devices use Ctrl-click. Remember to hold the Ctrl-key if you want to add a device later.

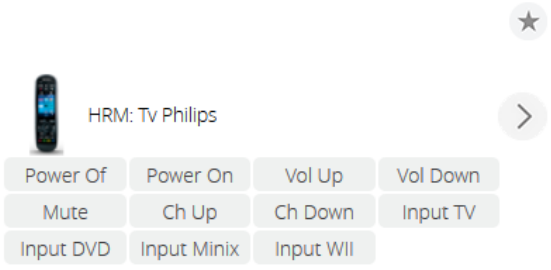
When you have configured a Philips Hue bridge on your Harmony, any configured lamps will show in the second drop down. Select the lamps you want to control. The lamps will have a matching Light device created.

When `Create child devices embedded` is yes, the child devices will show in the same room as the main Harmony Hub Control device. When you change the room of a child device, the child device will get recreated again in the same room as the parent, and you will lose its button definitions! This will also happen when changing the `Create child devices embedded` setting.

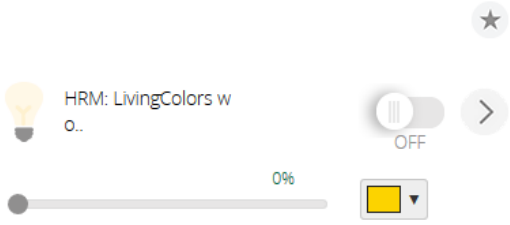
After clicking Save Changes, your Vera will reload several times. This is normal behavior when adding or removing child devices.

For each child device you can configure Buttons just as for the main device. Each Button is mapped to a command of the Harmony Device the child device is created for.





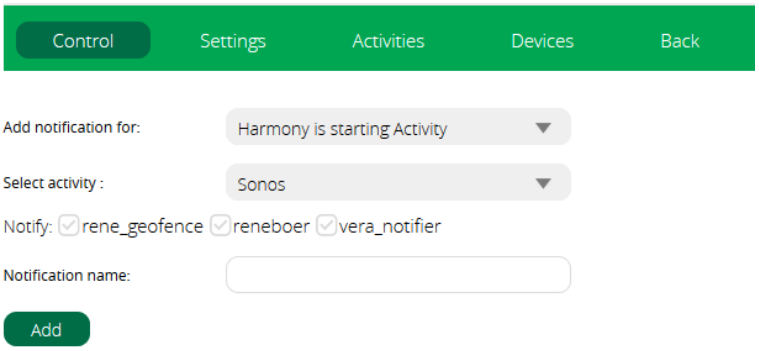
You can specify how many seconds the button should be held down. Click is just that, and a `Press` in seconds simulates holding the button for that number of seconds. The max is five seconds. This is normally more than sufficient for Volume Up or Down. Test the duration setting after changing it as it is difficult to estimate the exact effect.



This is an example of a Lamp device. The standard Vera light control definitions are used. This means you can even control them via the Vera mobile apps.

Configure Scenes Triggers and Notifications

Last option you have is to configure notifications or use the Harmony Hub Control devices to trigger scenes.

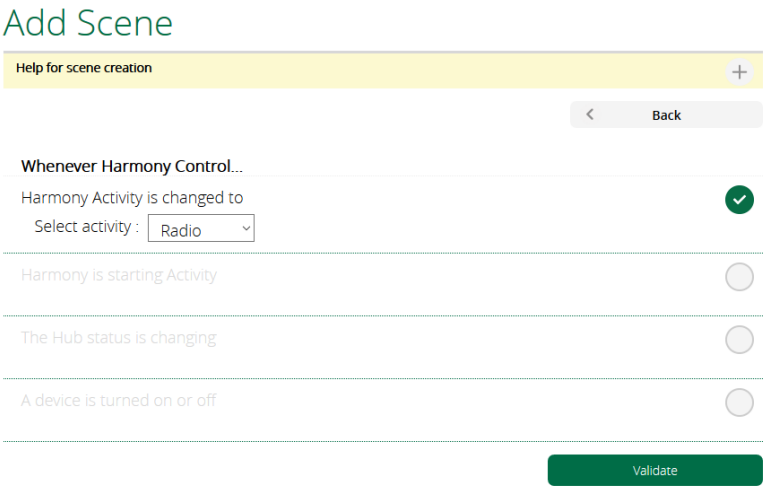


The triggers available are:

Harmony Activity is changed to: This will fire when the activity is completely started

Harmony is starting activity: This will fire when an activity is starting, but not yet done.

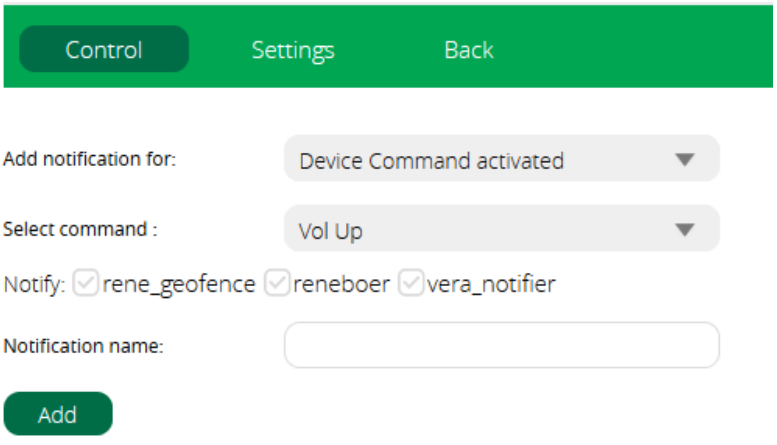
The Hub status is changing: When the hub has been turned off, after the PowerOff command, or activity ID = -1. When an Activity is starting on device, an activity is being started. When an Activity is active on device, all steps to start an activity have completed. Whenever the device is turning off, the PowerOff command is turning off all devices.



For the main harmony Hub Control device, the trigger will occur when the selected activity is turned on. Either via clicking the Buttons, a Scene, a luup.call_action command or when you start the activity via your remote or mobile App.

A device is turned on or off: Whenever the Hub is turned on (activity is not PowerOff). Whenever the device is turned off (activity is PowerOff)

Notifications or triggers for the child devices will only happen via clicking the Buttons, a Scene, or a luup.call_action command.



Control Polling the Hub

Using the standard settings the connection between your Vera (or openLuup device) and the Hub is kept open. This is required to assure any changes initiated by the Harmony remote or mobile app is correctly reflected on your Vera. If for some reason the connection lost, the plugin will try to reconnect every five minutes until it succeeds.

However, there may be times you do not want this. For example, if you turn off your Hub in Away or Vacation mode.

For this use the SetHubPolling action. Turn it off when needed. It will still be possible to use the plugin and send commands to your Harmony Hub, however you will miss status updates. Note that the polling will only be stopped when the Hub is off, that is, the CurrentActivityID is -1.

When turning it on again, the constant connection is restored and the plugin is fully functional again.

Turn off polling;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1", "
  SetHubPolling",{newPollingFlag="0"},28)
```

Turn on polling;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1", "
  SetHubPolling",{newPollingFlag="1"},28)
```

Detailed information on starting activities

With version 3.3 you have the possibility to closely watch activity changes with three new variables; StartingActivityID, StartingActivityStep and activityStatus.

StartingActivityID will hold the activity being started. It will match CurrenActivityID once the activity if fully started. You can use this value as a scene or notification trigger by selecting "Harmony is starting Activity". For very long running activities it can give a couple of seconds heads up of what activity is being started.

An starting an activity is basically a sequence of device commands send by you Hub to your devices as you have configured these in the MyHarmony app. You can follow these steps being executed by watching the StartingActivityStep variable. The of the variable format is ddd,n,t. Where ddd is the device a command is sent to, n is the steps number and t is the total number of steps executed by the activity.

You can find details for each activity by running the GetConfig command in your browser as explained below. This will give a large JSON file that is not so easy to read. Paste it into jsonlint (https://jsonlint.com) to format it. The steps would be the turning on the devices part of the activity, turning off any on devices that are not, the powerFeatures steps for each device and the enterActions for the activity. Bit of a puzzle as you can see.

Note that all three variables will also be populated when starting an activity from the Harmony remote or mobile app, provided SetHubPolling is on (1).

Last variable is activityStatus. This has four possible values as follows; 0 = Hub is off, 1 = Activity is starting, 2 = Activity is started, 3 = Hub is turning off. You can also use this as a scene or notification trigger "The Hub status is changing".

LUUP.Call_action commands

In LUA you can use the following luup.call_action commands in scenes LUUP code or your own plug ins:

GetCurrentActivityID, StartActivity, PowerOff, IssueDeviceCommand, SendDeviceCommand, ChangeChannel, SetHubPolling, FindActivityByID, FindActivityByName, FindDeviceByID, FindDeviceByID, ForceUpdateConfiguration, SetTarget.

When starting an activity or sending a command any set notifications or Scene trigger will fire. Most commands are clear by the examples below.

SetHubPolling (newPollingFlag: “0” or “1”) determines is the WebSocket is kept open (1) or re-opened and closed for each command (0). With the latter the plug-in will not get instant updates and notification messages from the Hub and should only be used if you plan to turn off the power to the Hub. Calling SetHubPolling with 1 after restoring power will re-enable the instant updates. If SetHubPolling is 1 and the connection to the Hub is lost, the plug-in will try to restore the connection after five minutes.

Some luup.call_action examples:

Get the current activity ID;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
"GetCurrentActivityID",{},28)
if (rc==0) then curActID = rarg.currentActivityID end
```

Start activity with ID 10439301;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
  "StartActivity",{newActivityID=10439301},28)
```

Start activity with name Watch TV;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
  "StartActivity",{newActivityID=”Watch TV”},28)
```

Send Mute command to device 21447853;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
  "IssueDeviceCommand",{DeviceID=21447853,Command="Mute"},28)
```

Send Mute command to device Radio;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
  "IssueDeviceCommand",{DeviceID=”Radio”,Command="Mute},28)
```

or Send a VolumeUp command with holding the button for five seconds to child device. #30 in this example;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:HarmonyDevice1",
  "SendDeviceCommand",{Command="VolumeUp",Duration="5"},30)
```

Turn on the default activity;

```
rc, rs, jb, rarg = luup.call_action("urn:upnp-org:serviceId:SwitchPower1","SetTarget",
  {newTargetValue=1},28)
```

Turn all activities off;

```
rc, rs, jb, rarg = luup.call_action("urn:upnp-org:serviceId:SwitchPower1","SetTarget",
  {newTargetValue=0},28)
```

Find activity ID with name Watch TV;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1",
  "FindActivityByName",{ActivityID=”Watch TV”},28)
```

Turn off polling;

```
rc, rs, jb, rarg = luup.call_action("urn:rboer-com:serviceId:Harmony1", "
  SetHubPolling",{newPollingFlag=”0”},28)
```

HTTP Request Commands

When the option Enable HTTP Request handler is set to Yes in the plug-in settings you can issue HTTP commands to your Vera as a proxy to control your Hub.

Use the http commands get_config, list_activities, list_devices, list_lamps, list_device_commands, update_config, get_current_activity_id, power_off, start_activity, change_channel, find_activity_by_name, find_activity_by_id, find_device_by_name, find_device_by_id, as described next to find out all arguments you can use with these functions.

When starting an activity or sending a command any set notifications or Scene trigger will fire.

Replace 192.168.1.1 with your Vera IP address and 66 your main plug-in ID:

To List all activities;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=list_activities
```

To List all devices as a JSON string;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=list_devices
```

To List all device commands for device 21133494;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=list_device_commands&cmdp1=21133494
```

To start the activity controlled by Button 1;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=start_activity&cmdp1=1
```

To start the activity 10439329;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=start_activity&cmdp1=10439329
```

To start the activity Watch TV;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=start_activity&cmdp1=”Watch TV”
```

To turn the Hub off;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=power_off
```

To get the current Activity ID;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=get_current_activity_id
```

To issue a device command for device holding button for five seconds;

```
http://192.168.1.1/port_3480/data_request?id=lr_Harmony66&cmd=issue_device_command&cmdP1=21447853&cmdP2=VolumeUp&cmdP3=5
```