

Remote Builder Help
Version 3.0.0
(11/06/24)

Table of Contents

Table of Contents	2
Section 1 - Introduction and Installation	3
Introduction	4
What are Remotes	5
Why Use Remotes.....	5
A Few Scenarios	5
How Remote Builder Works.....	6
Remote Builder Installation	7
Endpoints	7
Licensing.....	8
Section 2 - Modules	9
Fixed 6 Button Remote	10
Accessing the Remote.....	13
Logging	14
Custom 6 Button Remote	15
Overview of 6 Button Custom Remote	15
Remote Customization Options.....	16
Button Groups.....	16
Other Settings	17
ToolTip Selector	17
Synthetic Commands	18
TV Remote.....	19
Requirements.....	20
Create a New TV Remote.....	20
Select TV Device.....	21
Endpoints	21
Customize TV Remote.....	22
TV Remote Fixed Buttons.....	22
TV Remote Custom Buttons.....	22
Other Commands.....	23
Publish TV Remote	23

Remote Builder Version 3.0

Roku Remote.....	24
Create Roku Remote	24
Customize Remote	25
Fixed Buttons	25
Custom Buttons	25
Keypad	26
Input Handling.....	26
Publish Keypad	27
QR Code	28
It's a Wrap.....	29

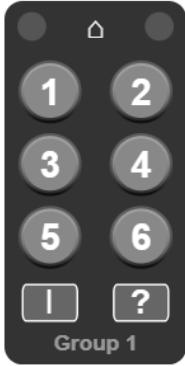
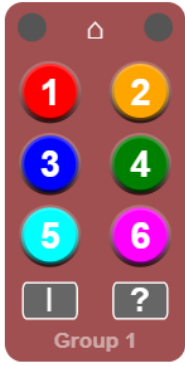

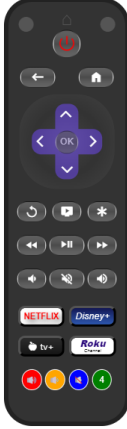











Section 1 - Introduction and Installation

Remote Builder Version 3.0

Introduction

Remote Builder allows you to create small but powerful “remote control” packages that can be placed on a dashboard or opened directly in the browser. The current range of remotes is shown in the table below.

Available Remotes

Fixed 6 Button Remote	Custom 6 Button Remote	TV Remote																								
																										
ROKU Remote	Keypad	QR Code																								
																										
Lighting Table																										
<table border="1"><thead><tr><th><input type="checkbox"/></th><th>Name</th><th>↻</th><th>State</th><th>Color</th><th>Dim/°K</th></tr></thead><tbody><tr><td><input type="checkbox"/></td><td>Office Left</td><td></td><td><input checked="" type="checkbox"/></td><td></td><td>96% <input checked="" type="checkbox"/></td></tr><tr><td><input type="checkbox"/></td><td>Office Right</td><td></td><td><input checked="" type="checkbox"/></td><td></td><td>96% <input checked="" type="checkbox"/></td></tr><tr><td><input type="checkbox"/></td><td>Virtual Bulb</td><td></td><td><input checked="" type="checkbox"/></td><td></td><td>32% <input type="checkbox"/></td></tr></tbody></table>			<input type="checkbox"/>	Name	↻	State	Color	Dim/°K	<input type="checkbox"/>	Office Left		<input checked="" type="checkbox"/>		96% <input checked="" type="checkbox"/>	<input type="checkbox"/>	Office Right		<input checked="" type="checkbox"/>		96% <input checked="" type="checkbox"/>	<input type="checkbox"/>	Virtual Bulb		<input checked="" type="checkbox"/>		32% <input type="checkbox"/>
<input type="checkbox"/>	Name	↻	State	Color	Dim/°K																					
<input type="checkbox"/>	Office Left		<input checked="" type="checkbox"/>		96% <input checked="" type="checkbox"/>																					
<input type="checkbox"/>	Office Right		<input checked="" type="checkbox"/>		96% <input checked="" type="checkbox"/>																					
<input type="checkbox"/>	Virtual Bulb		<input checked="" type="checkbox"/>		32% <input type="checkbox"/>																					

In this document you will find a section on each of these types of remote. In most cases their operation is pretty straightforward but there are some nuances it might help to be aware of.

Remote Builder Version 3.0

What are Remotes

Remotes are applets that can be embedded in a Dashboard or launched directly on a phone or tablet. Some are simply the digital equivalent of the ubiquitous remote controls that everyone has around their home for TV, audio, fans, lights, climate, garage doors etc. But unlike hardware based remote controls these digital equivalents can be customized in terms of their function and appearance.

Just like most physical remotes, many of these virtual remotes do not display status. They are generally intended for use within the home where the user will know if the requested action was performed or not, without the need for displaying status.

Why Use Remotes

Remotes solve several problems experienced in the Hubitat environment.

- 1) Simple interface for discrete tasks. Hubitat dashboards can be a bit intimidating for non-technical people, but everyone knows how to use a remote control.
- 2) Delegation of authority. By programming a remote to perform a discrete set of tasks you can delegate authority over those devices, but only for the specified functions.
- 3) No Hubitat app or dashboard required. These remotes can load directly within the browser and will scale to fit the device.
- 4) A customizable interface. You can change the icons, icon colors and background colors used on the remote to your own liking.
- 5) Tooltips. Each button has a customizable tooltip which indicates the function of the button.
- 6) Density. You can have a lot more information in a small space

A Few Scenarios

Your kids have some smart devices in their rooms, and you want an easy way to give them control over just those devices.

You have guests coming to stay and your guest bedroom has some automated elements. You can make a remote for changing the lights, fan, temperature, blinds, TV or whatever else you might have and share the cloud endpoint with your guests. They do not even need to access your WiFi.

You have a cat\dog sitter that comes over to feed or walk your animals. You could make a remote that allows them to unlock the front door, disable the alarm and turn on the light when they arrive and then reverse it all when they leave.

You have some physical button remotes around the home. Now you can add a virtual remote that produces the exact same actions including the double click or long press

Your dashboard is cluttered with controls and they take up too much space, or you just want to spruce up your dashboard with some better looking controls.

Remote Builder Version 3.0

How Remote Builder Works

There are nine components to Remote Builder of which two are required elements and the rest are optional. Some are only available if you have the Advanced Version of Remote Builder.

Required Components

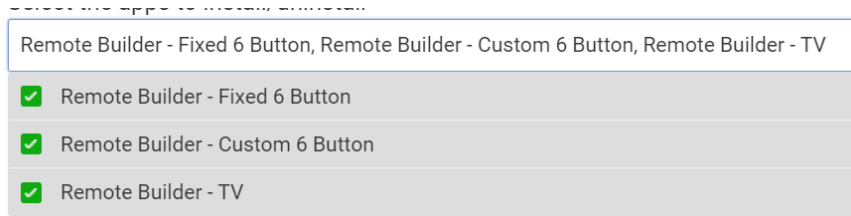
- 1) **Remote Builder Parent App**
- 2) **Remote Builder Storage Driver** – Device driver used for storing Remotes data.

Optional Modules (Child Apps)

- 3) **Fixed 6 Button Remote** – A 6 button whose actions can be customized but whose appearance cannot.
- 4) **Custom 6 Button Remote*** – A 6 button remote with the 3 button groups (18 buttons total) whose appearance can be customized.
- 5) **TV Remote*** - A 29 button TV remote including 10 customizable buttons that can point to unique functions or even to entirely different devices.
- 6) **Roku Remote** – A 25 button remote that looks like a Roku and works with the Roku device driver.
- 7) **Keypad** – A numeric keypad that can be used to submit codes which can in turn perform operations such as lock\unlock a door.
- 8) **QR Code** – Generates a QR Code for a URL that can be placed on a dashboard and is an easy to replicate one of these remotes onto another device.
- 9) **Lights*** – Generates a table of lights with all appropriate controls. These lights can be modified individually or in groups.

Modules marked * are only available in the Advanced version of Remote Builder.

When you install Remote Builder from HPM be sure to select the modules you want installed!



The Remote Builder parent app is the primary organizing app.

<input type="checkbox"/>	Remote Builder	Remote Builder (user)
<input type="checkbox"/>	Basement	Remote Builder - TV (user)
<input type="checkbox"/>	Dawn	Remote Builder - Custom 6 Button (user)
<input type="checkbox"/>	Dusty	Remote Builder - Custom 6 Button (user)
<input type="checkbox"/>	Gary	Remote Builder - Custom 6 Button (user)
<input type="checkbox"/>	Heather	Remote Builder - Custom 6 Button (user)
<input type="checkbox"/>	Living Room	Remote Builder - TV (user)
<input type="checkbox"/>	Office TV	Remote Builder - TV (user)
<input type="checkbox"/>	Sprinklers	Remote Builder - Fixed 6 Button (user)

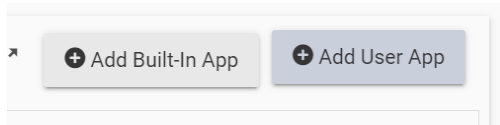
Remote Builder Version 3.0

Remote Builder Installation

Remote Builder is listed in Hubitat Package manager. Choose to install by tags and select the **Dashboards** tag. Select **Remote Builder for Hubitat** and complete the installation process. This will place the code on your hub and then there are a few steps to complete the installation.

Tile Builder for Hubitat by Gary J. Milne

Create dashboard tiles that are highly customizable and can contain data from multiple devices.



1. Go to the Apps tab and click on **Add User App**
2. Select **Remote Builder** from the list of available apps.
3. **Remote Builder** will install and bring you to the parent screen.
4. **Select the appropriate License Type (see paragraph below on licensing).**
5. Under **Device Creation** you must first create the storage device and then connect the app to the device. Just use the default device for now.

✗ - A Remote Builder Storage Device is not connected.

Select a Remote Builder Storage Device

Remote Builder Storage Device 1

Create Device

Connect Device

Delete Device

You must connect to a storage device in order to publish tiles.

Next ▶

6. Once the device is created and connected it will look like this.

✔ - Remote Builder Storage Device 1 is connected.

You have successfully connected to a Remote Builder Storage Device on your system.

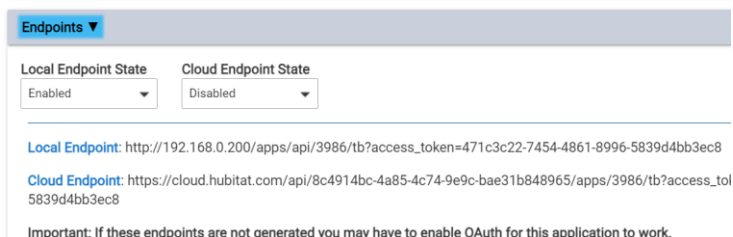
Disconnect Device

Next ▶

7. We can now create our first remote.

Endpoints

Hubitat uses endpoints to publish information both within the local network and to the cloud. Remote Builder uses these endpoints to publish these remotes, so they are accessible to anyone with the network address. Each Remote Builder module has an Endpoints section where the local and cloud endpoints are available. Endpoints can also be enabled or disabled here.



Remote Builder Version 3.0

Licensing

Remote Builder has Standard and Advanced versions just like **Tile Builder** and shares the same key. If you already have a Tile Builder key you can use it on your Remote Builder installation to unlock the Advanced version.

The table below shows which version of Remote Builder is required for each module.

Module Name	Standard	Advanced
Fixed 6 Button Remote	Yes	Yes
Custom 6 Button Remote	-	Yes
TV Remote	-	Yes
Roku Remote	Yes	Yes
Keypad	Yes	Yes
QR Code	Yes	Yes
Lights	-	Yes

To obtain an Advanced license for Remote Builder and Tile Builder go to the licensing section in the parent app and follow the instructions. When you purchase a license, it allows me to work on developing new and improved modules for the betterment of Hubitat vs working a part time job to generate the same income.

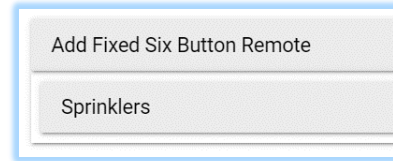
If you don't want to pay for an Advanced license you are still free to use the Standard version as much as you wish.

Section 2 - Modules

Remote Builder Version 3.0

Fixed 6 Button Remote

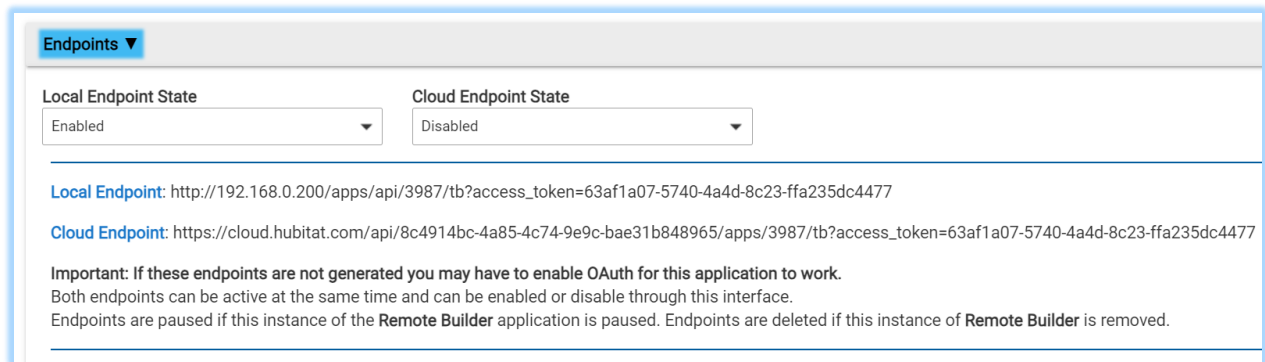
Within the **Remote Builder** parent app go to the section called **Create \Edit Remotes** and select Add **Fixed Six Button Remote**. The **Fixed 6 Button Remote** main screen will be displayed.



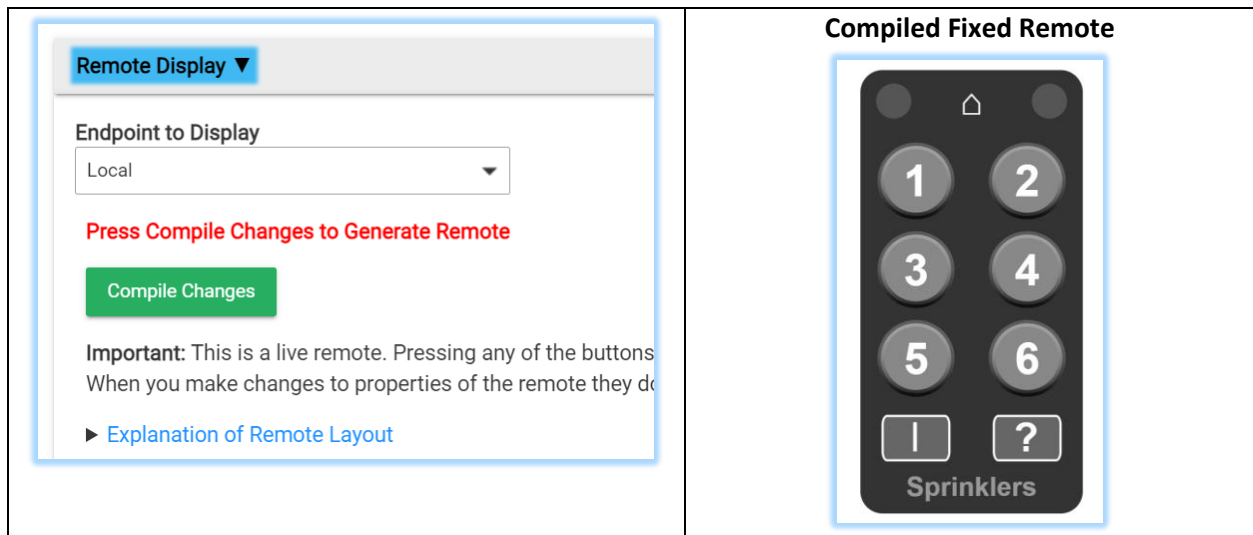
The app display is divided into 5 sections.

Introduction: A brief description of the program you are running and what it can do.

Endpoints: Where the generated remote can be accessed as well as being enabled\disabled.



Remote Display: This is where the generated remote can be previewed and tested. The actual remote is stored in a compiled form for efficiency. When changes are made, they must be compiled to take effect.



Remote Builder Version 3.0

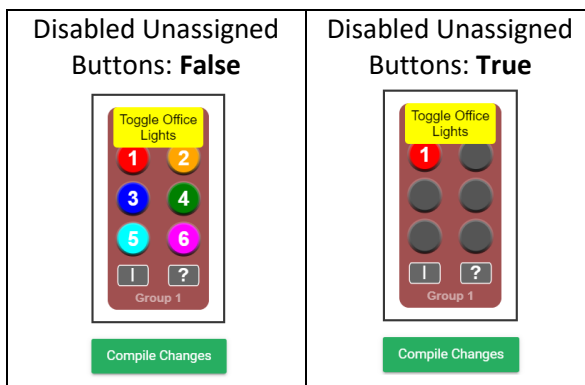
Customize Display: This is where you select the device and actions that will be generated by a button press as well as the Tooltip that is displayed.

The 'Customize Remote' interface shows a configuration panel for 'Group 1'. It includes a 'Group 1 Background Color' selector, a 'Group 1 Title Text' field set to 'Group 1', and two button configuration sections. 'Button 1 Device' is set to 'Office Lights - Gary', 'Command' is '*toggle', and 'Tooltip (optional)' is 'Toggle Office Lights'. 'Button 2' is currently unconfigured.

After a device\action is selected it looks like this.

This close-up shows the configuration for 'Button 1'. The device is 'Office Lights - Gary', the command is '*toggle', and the tooltip is 'Toggle Office Lights'.

After you Compile the remote it looks like this when you mouse over button 1.



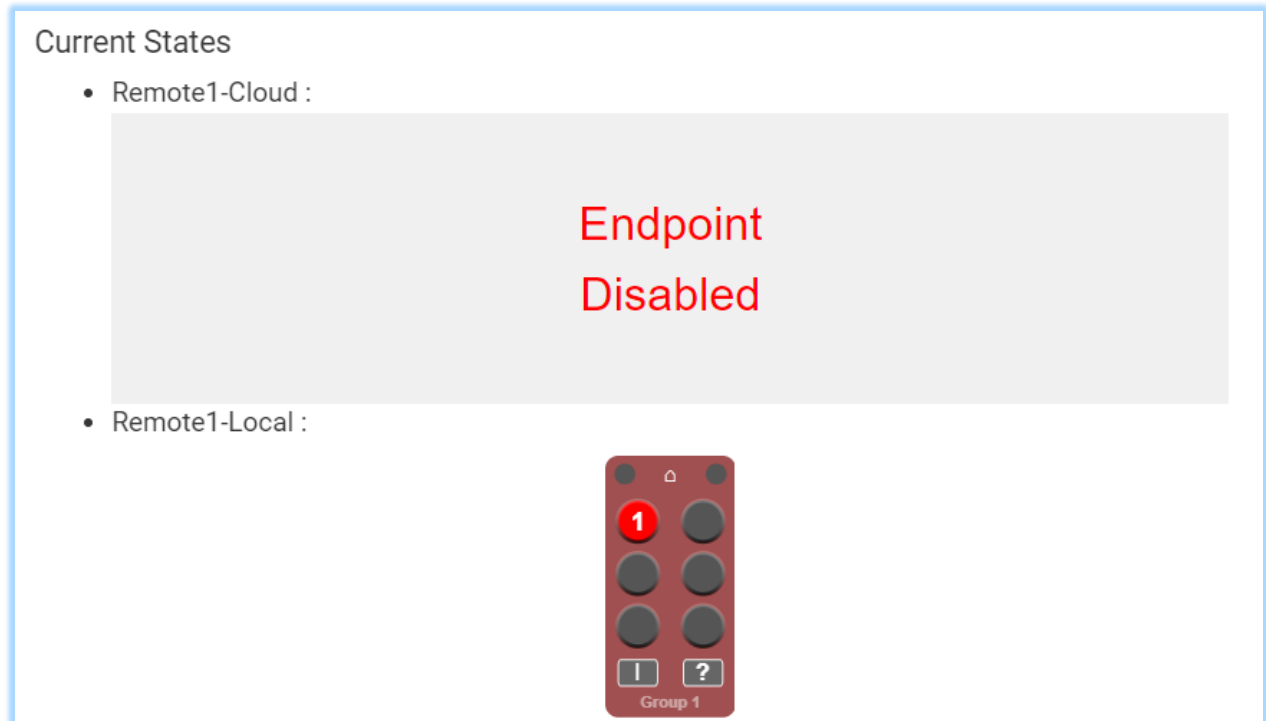
Publish Remote: If you are familiar with Tile Builder you will know that publishing the output to a tile is a necessary step to viewing the output. With **Remote Builder** publishing a remote to an attribute is optional, this is because Remotes work perfectly well with just the Endpoint address.

To publish a remote, select the attribute prefix to publish it under. Attributes begin with the prefix Remote1 – Remote25. You can then click on the Publish Remote button.

The 'Publish Remote' interface includes fields for 'Attribute to store the Remote?' (set to 'Remote1'), 'Name this Remote*' (set to 'Lights'), and 'For Reference Only: Remotes in Use' (set to 'Click to set'). A 'Publish Remote' button is located at the bottom.

Remote Builder Version 3.0

If we look within the **Remote Builder Storage Device** we will find that the attributes **Remote1-Local** and **Remote1-Cloud** are populated with links to the remote that we generated.



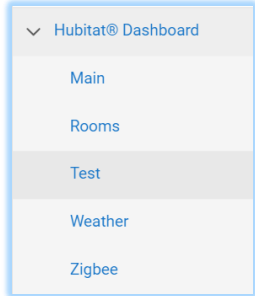
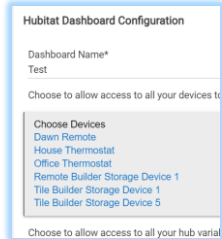

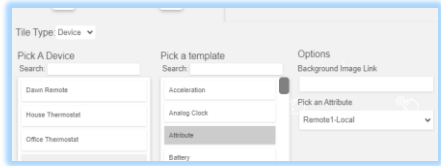


Note: This is a live preview of the Endpoint. This remote is active and can be clicked on to execute commands so be careful where you click.

Remote Builder Version 3.0

Accessing the Remote

There are 2 typical ways to access the remote. The first is by using the links provided in the Endpoints section. These can be stored as favorites or shared via email or text.

The other method is to add the remote device to a Hubitat Dashboard. This is how you do that.

Go to the desired Hubitat Dashboard entry listed under Apps. It will look something like this.	
Now add the Remote Builder Storage Device to your list of allowed devices so it looks like this.	
Now go to the Dashboard you selected and click the + sign to add a device.	
Pick the Remote Builder Storage Device. Select attribute and then the Remote you wish to publish. In this case it is Remote1-Local .	
Once you have done that it will appear on the Dashboard in the position and size you have selected. Remotes are built with Vector Graphics so they can be scaled to whatever size you wish.	
If the Endpoint is disabled it will look like this. You can enable or disable the Endpoint using the Remote Builder App instance for this remote.	

Remote Builder Version 3.0

Logging

Under the section **More Options** you can enable\disable the logging of connections to the Endpoint as well as actions initiated by the remotes.

☒ More Options

In this section you can enable logging of any connection and action requests received.
You can also rebuild the endpoints if you choose to refresh the OAuth client secret

☐ Enable Debug logging?

☒ Log errors encountered?

☐ Record All Connection Requests?

☐ Record All Action Requests?

Rebuild Endpoint(s)

With “Record All Action Requests” enabled the log will look something like this.

Dawn	Dawn Remote	Dawn Remote	Bedroom Evening Bright	Bedroom Fan Lights	MB Fan Bulb 2	Bedroom Fan Lights Off	MB Fan Bulb 3
app:3927	2024-08-14 09:04:56.026 AM	info	Remote Builder Data Received - Remote: 1 - Name: Dawn - Button: 6 - Device: Dawn Remote - Command: *doubleTap3				
app:3927	2024-08-14 09:04:53.606 AM	info	Remote Builder Data Received - Remote: 1 - Name: Dawn - Button: 5 - Device: Dawn Remote - Command: *push3				
app:3927	2024-08-14 09:04:44.336 AM	info	Remote Builder Data Received - Remote: 1 - Name: Dawn - Button: 2 - Device: Dawn Remote - Command: *doubleTap1				
app:3927	2024-08-14 09:04:41.397 AM	info	Remote Builder Data Received - Remote: 1 - Name: Dawn - Button: 1 - Device: Dawn Remote - Command: *push1				

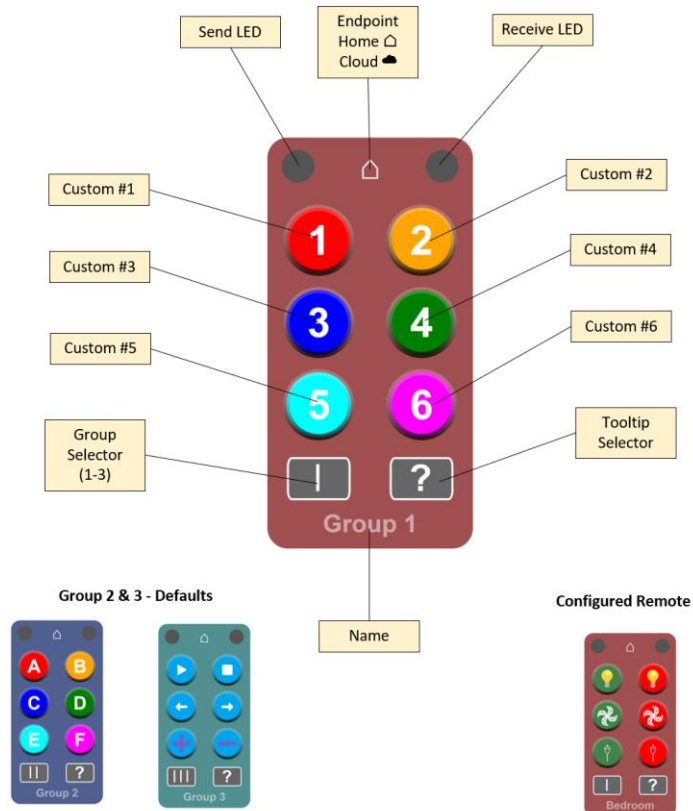
Debug logging is really intended for use by the developer or at the request of the developer in a troubleshooting scenario.

Custom 6 Button Remote

This section assumes you have already read the **Creating a Fixed Six Button Remote** section and only discusses those parts of the process which are different.

As the title suggests, this type of remote provides a lot more customization options than its **Fixed** sibling. The custom remote also offers one other big advantage, which is that it supports 3 button groups effectively making it an 18-button remote.

Overview of 6 Button Custom Remote



Remote Builder Version 3.0

Remote Customization Options

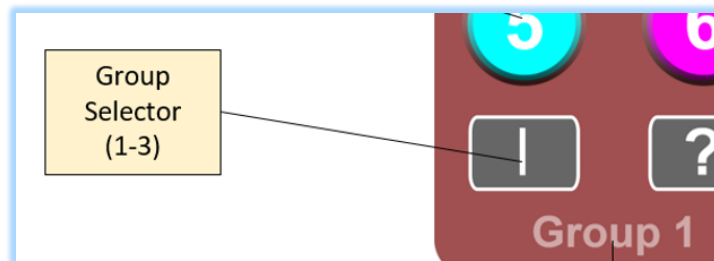
The diagram below illustrates the various customization options available for each button group.

The screenshot shows the 'Customize Remote' interface. At the top, there's a 'Select Button Group to Customize' dropdown set to 'ONE'. To its right is a 'Group 1 Background Color' color bar and a 'Group 1 Title Text' field containing 'Bedroom'. Below these are six rows, each representing a button device. Each row has a 'Button X Device' label (e.g., 'Button 1 Device'), a 'Command' dropdown, a 'Button Color' color bar, a 'Character*' field with a lightbulb icon, a 'Text Color' field, and a 'Tooltip (optional)' field. The buttons are labeled 'Dawn Remote'. The 'Disable Unassigned Buttons' dropdown at the bottom is set to 'False'.

Button Device	Command	Button Color	Character*	Text Color	Tooltip (optional)
Button 1 Device Dawn Remote	*push1	Green	?		Light On\Cycle
Button 2 Device Dawn Remote	*doubleTap1	Red	?		Light Off
Button 3 Device Dawn Remote	*push2	Green	?		Fan On\Cycle
Button 4 Device Dawn Remote	*push2	Red	?		Fan Off
Button 5 Device Dawn Remote	*push3	Green	?		Lamp On\Cycle
Button 6 Device Dawn Remote	*doubleTap3	Red	?		Lamp Off

Button Groups

A Custom 6 Button Remote has three button groups for a total of 18 buttons. You can toggle through these three button groups using the Group Selector button as shown below.



Other Settings

You can change the background color for each button group and give it a name.

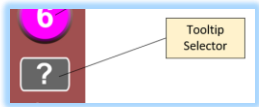
Group 1 Background Color	Group 1 Title Text
<div></div>	Bedroom

Disable Unassigned Buttons You can choose to disable those buttons that do not have a valid device\command assigned.	<div>Disable Unassigned Buttons</div> <div>False</div>
In this case none of the buttons on Group 3 are assigned so they all appear greyed out.	<div><div>Group 3</div></div>

ToolTip Selector

When using a device that has a pointer device such as a mouse, placing the cursor over one of the action buttons will cause the assigned tooltip to be revealed. This acts as a reminder of what action will be initiated by pressing the button.

But most phones and tables do not have a pointing device that can be used to reveal the Tooltip. For touch devices the Tooltip can still be revealed using the Tooltip Selector button shown below.



The Tooltip Selector acts as a toggle. The picture above shows it in it's normal state. When it is active the question mark is displayed in green. Once the Tooltip Selector is activated, clicking on any of the buttons will display the tooltip, but does not invoke the assigned action. The Tooltip selector remains in this state until it is toggled off again.

Remote Builder Version 3.0

Synthetic Commands

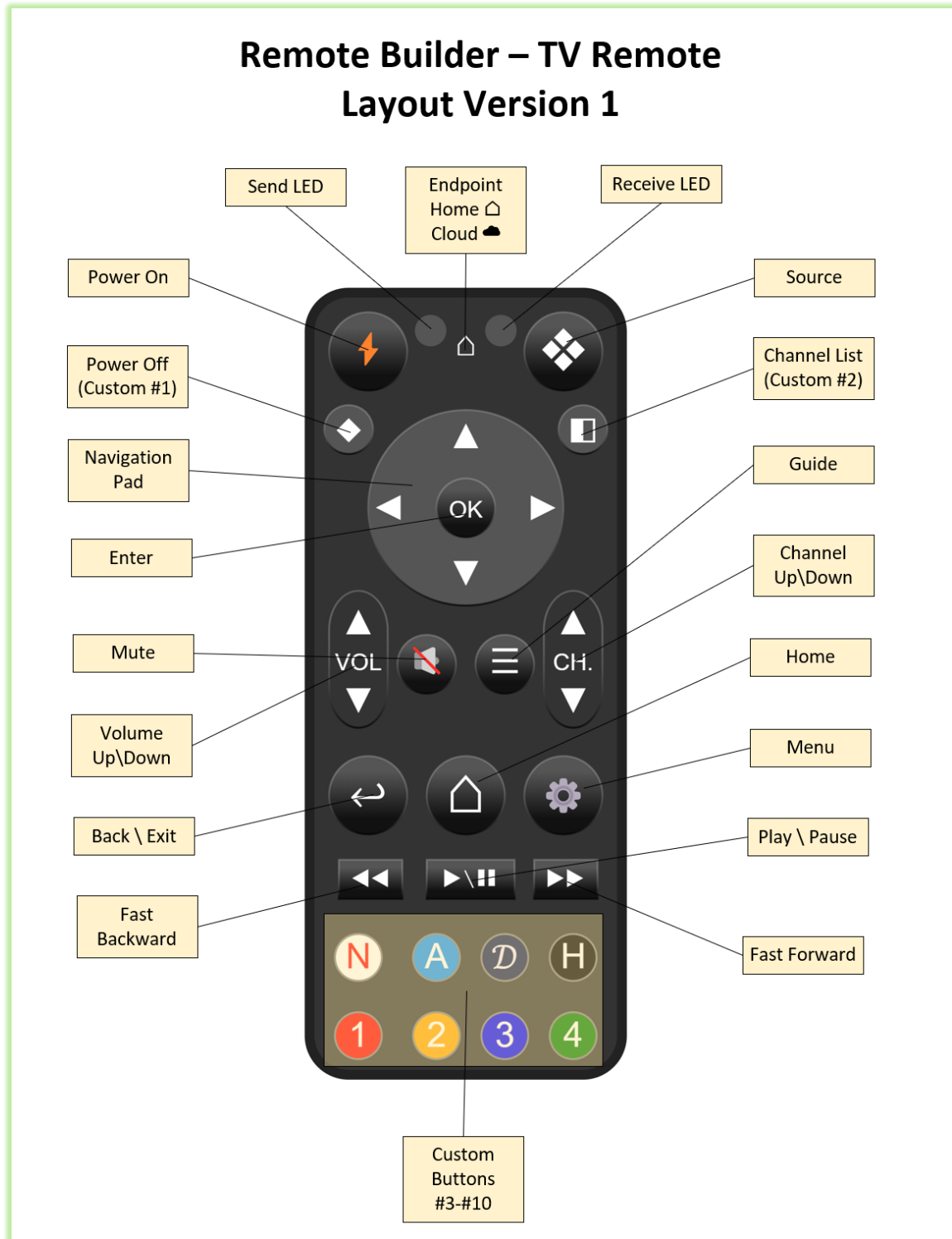
If you have tried out Remote Builder you may have noticed that some of the commands begin with an * and may look like *toggle as in the example above.

If remote builder sees a device with On\Off commands but it does not have a toggle command then it inserts a synthetic toggle into the available command list and prefixes it with an * so it appears at the top. The result is that a single button can be used to turn any device on and off vs requiring two buttons, one to turn on and the other to turn off.

Remote Builder also adds synthetic commands for button controllers by adding *push1 - *push4 and *doubleTap1 - *doubleTap4. This allows these commands to be implemented via a single button push even though the commands push and doubleTap require a parameter.

TV Remote

The TV Remote is just a virtual rendition of a physical TV remote that we are all used to. The default layout is shown in the diagram below.



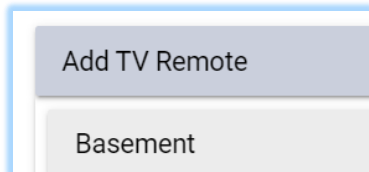
Remote Builder Version 3.0

Requirements

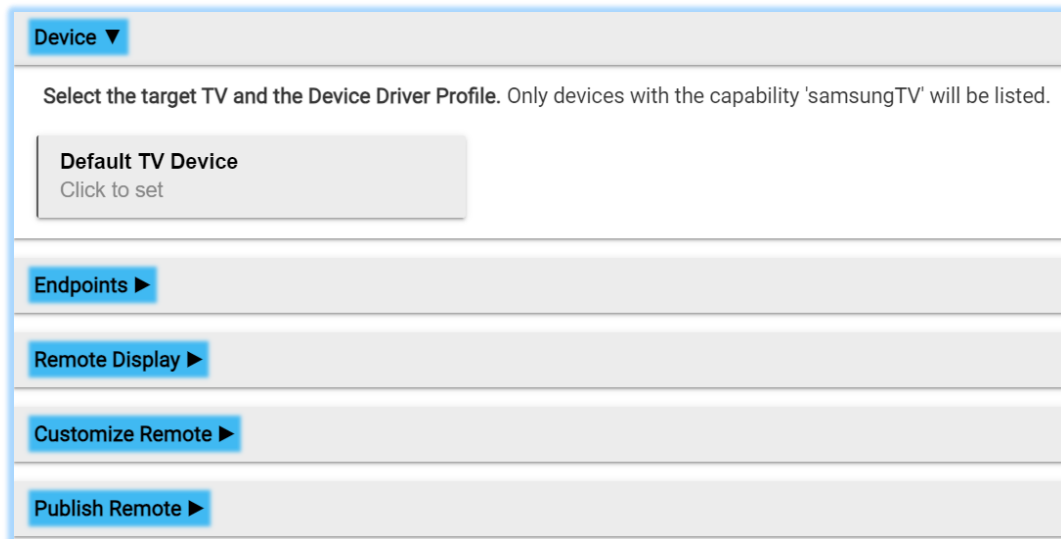
This remote is designed specifically to control a TV and can only operate in conjunction with a device driver that has implemented the SamsungTV capability. Before you can use this remote you must be sure that your Samsung TV is already part of your Hubitat device list and is responding to commands.

Create a New TV Remote

Go into the Remote Builder Parent App and click on **Add TV Remote** to create a new TV remote.



You will now see the app to configure a new TV remote which looks like this.

A screenshot of a configuration screen for a new TV remote. The screen has a light gray background with a blue border. At the top, there is a section titled 'Device' with a dropdown arrow. Below this, a text instruction reads: 'Select the target TV and the Device Driver Profile. Only devices with the capability 'samsungTV' will be listed.' Underneath the instruction is a button labeled 'Default TV Device' with the text 'Click to set' below it. Below the button are five horizontal sections, each with a blue header and a right-pointing arrow: 'Endpoints', 'Remote Display', 'Customize Remote', and 'Publish Remote'.

Remote Builder Version 3.0

Select TV Device

Select a TV device. If your list is empty, you don't have one configured within your environment and this TV Remote is of no use to you.

The only device driver supported on release is the Samsung TV Remote driver by David GutHeinz and it is automatically selected. Others will be added in future versions if they are adequately in demand and supported.

Click on **Apply Profile** to generate the initial remote. Your remote is now active using default settings.

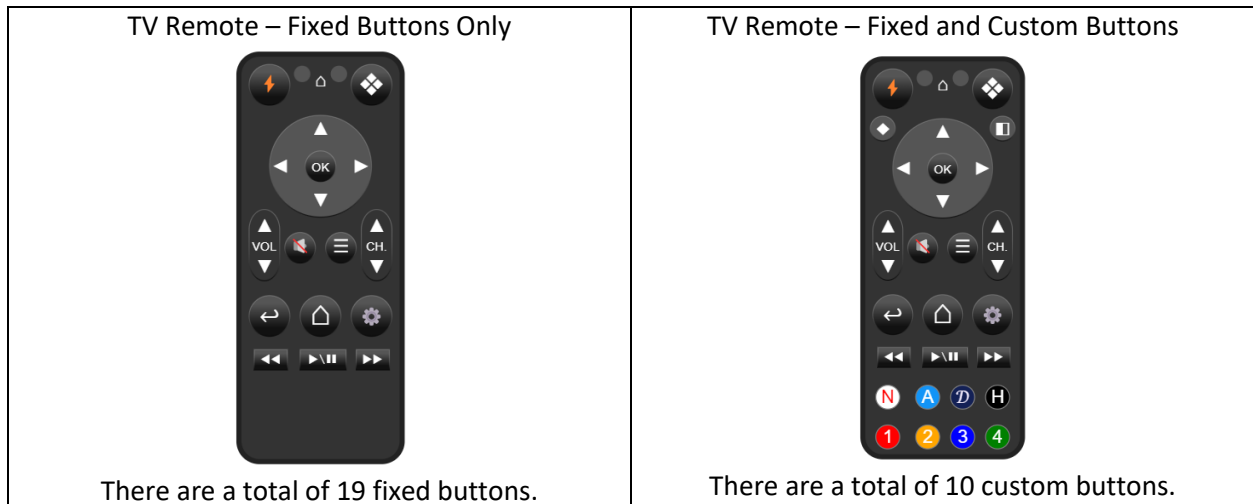
Endpoints

Your endpoints are automatically created and are accessible under the Endpoints section. By default only the local endpoint is enabled which means it is only accessible via your local network.

Remote Builder Version 3.0

Customize TV Remote

The TV remote has two types of buttons, Fixed and Custom. With Fixed buttons the only configurable option is to change the command being executed by that button press.



TV Remote Fixed Buttons

With Fixed buttons only the command to be executed can be changed.

Customize Remote

Select Button Group *
FIXED

Command (⚡) on	Command (⬅️) source	Command (⬆️) arrowUp	Command (⬇️) arrowDown	Command (⬅️) arrowLeft
Command (➡️) arrowRight	Command (OK) enter	Command (⬆️) volumeUp	Command (⬇️) volumeDown	Command (🔇) mute
Command (≡) guide	Command (⬆️) channelUp	Command (⬇️) channelDown	Command (⏮️) exit	Command (🏠) home
Command (⚙️) menu	Command (⏮️) fastBack	Command (▶️⏮️) play	Command (▶️▶️) fastForward	

TV Remote Custom Buttons

With Custom buttons multiple properties can be changed.

Customize Remote

Select Button Group *
CUSTOM

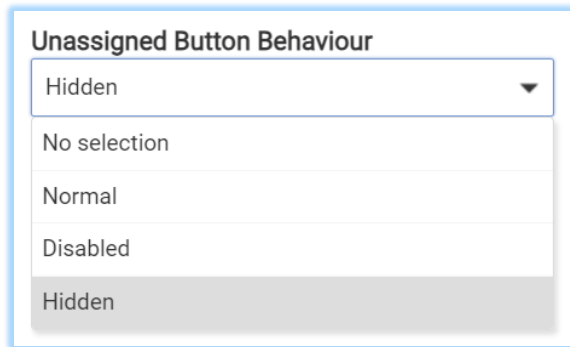
Custom Button 1 Device Office TV	Command off	Button Color	Character* ⬆️	Text Color
Custom Button 2 Device Office TV	Command channelList	Button Color	Character* 📺	Text Color
Custom Button 3 Device Office TV	Command appRunNetflix	Button Color	Character* N	Text Color
Custom Button 4 Device Office TV	Command appRunPrimeVideo	Button Color	Character* A	Text Color

Note: You can even select to execute different commands on devices other than the TV itself. For example, you could have a custom button that toggled the lights or closed the blinds.

Remote Builder Version 3.0

Other Commands

The unassigned button behavior has one additional property in the TV Remote which is to allow for unassigned custom buttons to be hidden vs just disabled. This gives the remote a cleaner look.



The image shows a dropdown menu titled "Unassigned Button Behaviour". The menu is open, displaying five options: "Hidden", "No selection", "Normal", "Disabled", and "Hidden". The first "Hidden" option is selected, and the second "Hidden" option at the bottom is highlighted with a grey background. A small downward arrow is visible on the right side of the first "Hidden" option.

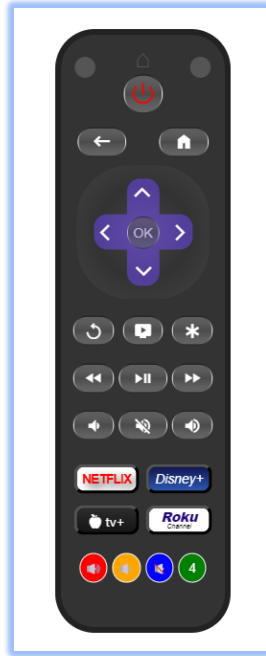
Publish TV Remote

Publishing the TV remote is the same as other Remote Builder modules. See the description under the Fixed 6 Button Remote for more information.

Remote Builder Version 3.0

Roku Remote

The Roku Remote is a virtual rendition of a physical Roku remote. The default layout is shown in the diagram below. If you have a Roku device the meaning of these buttons is self-explanatory except for the four custom buttons on the very bottom.



Create Roku Remote

When you create a Roku Remote you must select a Roku device from your Hubitat device list. If you do not have one you must install the Roku driver first and ensure that it is operational before continuing.

Default Roku Device
Office Roku

Select the Device Driver Profile

Roku Connect (>= 2.8.2) by Armand Welsh

Click **Apply Profile** after making your selections. The profile selected determines the mapping of key presses to the execution of commands. Applying a profile wipes out the current configuration of the remote including any custom buttons!

Apply Profile

This will apply all the default actions to the buttons on the virtual remote. There is nothing else that needs to be done. You can now access the remote via the links in the Endpoints section or you can publish the remote to a Hubitat dashboard using the **Publish Remote** section.

Remote Builder Version 3.0

Customize Remote

There are a number of customizations available for those that wish to go beyond the basics. Open the Customizations section.

Fixed Buttons

To modify any of the standard buttons select the **FIXED** Button Group where you can assign meaning to any of the buttons as shown below. The appearance of these buttons cannot be modified.

Select Button Group *	Show Parameters *	Commands Per Line *
FIXED	TRUE	2

Button (⏻)	Parameter	Button (←)	Parameter
keyPress	Power	keyPress	Back

Button (🏠)	Parameter	Button (^)	Parameter
home	?	keyPress	Up

Button (<)	Parameter	Button (>)	Parameter
keyPress	Left	keyPress	Right

Button (▼)	Parameter	Button (↩)	Parameter
keyPress	Down	keyPress	Select

Custom Buttons

The 4 custom buttons on the bottom are at your disposal to do whatever you want. They can be assigned to any device, not just the Roku to perform a variety of actions. You can also customize the appearance and content of these buttons as shown below.

Select Button Group *	Show Parameters *
CUSTOM	FALSE

Custom Button 1 Device Office TV	Command volumeUp	Button Color 	Character* 	Text Color <input type="text"/>
Custom Button 2 Device Office TV	Command volumeDown	Button Color 	Character* 	Text Color <input type="text"/>
Custom Button 3 Device Office TV	Command mute	Button Color 	Character* 	Text Color <input type="text"/>
Custom Button 4 Device Office Roku	Command Click to set	Button Color 	Character* 4	Text Color <input type="text"/>

Remote Builder Version 3.0

Keypad

The keypad allows you to access a virtual keypad as shown below and submit a series of numbers to Hubitat and then decide what to do with them.


There are no configuration options for the appearance at this time except for a title as shown below.

Endpoint to Display

Local

Title or blank for none

Back Door



Compile Changes

Input Handling

This section allows you to determine what happens when a series of numbers are submitted. There are 3 options.

1. Send the submitted values to a device as a parameter.

Input Handling

Send Data To:

Device

Device

Laundry Room 1

Command

setLevel

Note: The received code will be sent to the specified device\command as the first parameter.

2. Send the submitted values to a Hub String Variable. Only String variables will be selectable.

Send Data To:

Hub Variable

Hub String Variable

!Lock

Note: The received code will be sent to the specified Hub variable. You can use this method to trigger a rule by monitoring a Hub variable.

You can now implement a Rule to process this value whenever it changes and perform an action.

Remote Builder Version 3.0

3. Send the submitted value to a string comparison. If the values match, then the specified command will be executed. You can configure up to five unique commands to perform.

Send Data To:

Compare ▼

Compare 1

3451



Device 1

Office Left

Command 1

off ▼

Compare 2

6063



Device 2

Office Left

Command 2

on ▼

Compare 3

*



Device 3

Back Door

Command 3

lock ▼

Publish Keypad

You can access the virtual keypad directly via one of the endpoints or you can publish it to a Dashboard in the normal way as shown below.

Publish Remote ▼

Attribute to store the Remote? *

Remote1 ▼

Name this Remote*

Back Door



The **Remote Name** given here will also be used as the name for this instance. Only links to the Local and Cloud Endpoints are stored in the Remote Builder.

► [Publishing Notes](#)

Publish Remote

QR Code Generator

The QR Code generator is an easy way of putting a QR Code onto your dashboard. Someone with a mobile phone or tablet can scan that QR code and open whatever resource you have linked to.

The creation of a QR Code is quite simple. Enter the URL information and appropriate title into the boxes below.

Customize ▼

Enter URL for QR Code. Include http:// or https:// *

https://www.google.com

Select Size *

150 x 150 ▼

Show Title

True ▼

QR Code Title

Google Search

Title Text Color

Is Title Clickable

True ▼

Now go to the QR Code Display section and click Compile Changes and the QR Code will be generated. You can modify the color of the title text as well as make it clickable using the configuration options.

QR Code Display ▼

Endpoint to Display

Local ▼

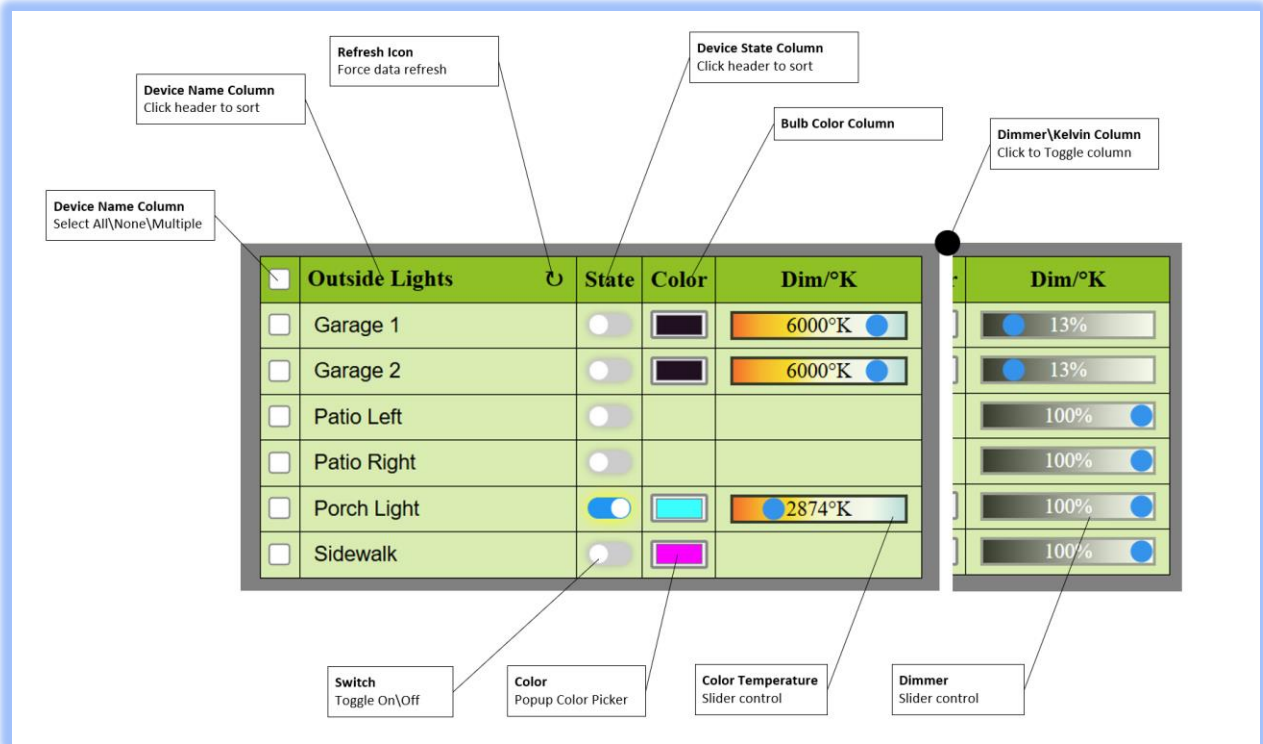


Compile Changes

You can now publish the QR Code to your dashboard in the normal way.

SmartGrid

SmartGrid combines the tabular format of Tile Builder with the ability to control certain elements as shown below.



It is primarily designed for use with bulbs as it exposes switch, color, dimmer and color temperature into a single table, but it can be used with any switchable device as you will see later. The ability to multi-select devices and change multiple devices at once makes it a convenient and compact way to work with groups of lights.

Create a New SmartGrid

After you have installed the SmartGrid child module you can select to “Add SmartGrid” from the parent.

First you must select the devices you wish to work with in your grid. If you wish you can select from a filtered list by selecting a required capability like “Color Devices” or “Dimmer”. Once you have selected devices the SmartGrid will be displayed and show all the current values. It will look like this:

Remote Builder Version 3.0

<input type="checkbox"/>	Name	↻	State	Color	Dim/°K
<input type="checkbox"/>	Garage 1		<input type="checkbox"/>		<input type="range" value="13%"/>
<input type="checkbox"/>	Garage 2		<input type="checkbox"/>		<input type="range" value="13%"/>
<input type="checkbox"/>	Patio Left		<input type="checkbox"/>		<input type="range" value="100%"/>
<input type="checkbox"/>	Patio Right		<input type="checkbox"/>		<input type="range" value="100%"/>
<input type="checkbox"/>	Porch Light		<input checked="" type="checkbox"/>		<input type="range" value="100%"/>

All devices will have a switch so the state column will always be fully populated, but the other columns will only have controls relevant to the device capability, such as color or dimmer.

Customizing a SmartGrid

The default settings provide an attractive interface, but you can change many of these settings if you choose. Customizations are performed through the “**Customize**” section as shown below.

Customize

Table

No selection

General

Table

Title

Headers

Rows

Columns

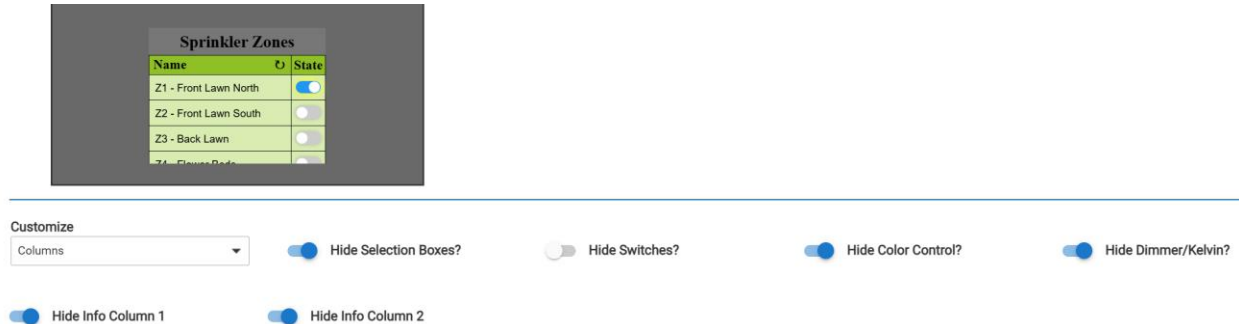
Info

Most of these are self-explanatory and you can experiment with them at your leisure but I will draw attention to a few concepts that are not present in Tile Builder tables.

Remote Builder Version 3.0

Hiding Columns

On the Customize\Columns menu you can choose to hide columns that you don't wish to appear on the final table. In the case below I have created a table for Sprinklers and have hidden the multi-select column (can only have one active zone at a time), Color and Dimmer/Kelvin columns as they do not apply.



Info Columns

Info columns let you add up to two extra columns to the table that show additional information about the device in that row. Info columns are available via the Customize menu under "Customize\Info".

Available info columns are as follows:

Name	Description
lastOn	The time at which the device was last in an On state.
lastOff	The time at which the device was last in an Off state.
lastOnDuration	The amount of time in dd:hh:mm(:ss) that the device was last in an On state. If the device is currently On this text will be green and the lastOnDuration is still incrementing. If it is off the text will be in red and the lastOnDuration is fixed.
lastOffDuration	The amount of time in dd:hh:mm(:ss) that the device was last in an Off state. If the device is currently Off this text will be red and the lastOffDuration is still incrementing. If it is On the text will be in black and the lastOffDuration is fixed.
roomName	This is the name of the room that the device is assigned to.
colorName	This is the colorName that is equivalent to the current color of the bulb.
colorMode	This indicates the color mode for any color bulb. These are typically RGB or CT but may be EFFECTS.
power	This indicates the amount of power the attached device is using if the device supports the power attribute.
deviceId	This is the 4-digit ID that Hubitat assigns to each device.

When a selected info attribute does not apply to a given device the field will display the "invalidAttribute" property. This can be changed in the **Customize\General** section to something of your choice.

Remote Builder Version 3.0

We can make our Sprinkler table more useful by adding **Last On** and **Duration** columns to the table so that I can check in on my scheduled watering.

Sprinkler Zones				
Name	🔄	State	Last On	Duration
Z1 - Front Lawn North	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	8:29 AM	1h 47m 7s
Z2 - Front Lawn South	<input type="checkbox"/>	<input type="checkbox"/>	5:00 AM	40m 0s
Z3 - Back Lawn	<input type="checkbox"/>	<input type="checkbox"/>	6:00 AM	9m 59s
Z4 - Flower Beds	<input type="checkbox"/>	<input type="checkbox"/>	7:00 AM	29m 59s
Z5 - Patio West	<input type="checkbox"/>	<input type="checkbox"/>	8:00 AM	20m 0s
Z6 - Patio East	<input type="checkbox"/>	<input type="checkbox"/>	8:30 AM	19m 59s

Sprinkler Zones				
Name	🔄	State	Last On	Duration
Z1 - Front Lawn North	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Wed @ 8:29 AM	2h 10m 48s
Z2 - Front Lawn South	<input type="checkbox"/>	<input type="checkbox"/>	Wed @ 5:00 AM	40m 0s
Z3 - Back Lawn	<input type="checkbox"/>	<input type="checkbox"/>	Wed @ 6:00 AM	9m 59s
Z4 - Flower Beds	<input type="checkbox"/>	<input type="checkbox"/>	Wed @ 7:00 AM	29m 59s
Z5 - Patio West	<input type="checkbox"/>	<input type="checkbox"/>	Wed @ 8:00 AM	20m 0s
Z6 - Patio East	<input type="checkbox"/>	<input type="checkbox"/>	Wed @ 8:30 AM	19m 59s

Note: The default format for the dates\times and duration can be changed in the **Customize\General** section so that it could look like the version in the second column of the above table.

Publishing

With Remote Builder publishing is only required if you are going to place the SmartGrid onto the dashboard. To publish a dashboard simply select an attribute to store the link and give the remote a name as shown below and click on the **Publish and Subscribe** button.

Publish Remote ▼

Attribute to store the Remote? (Optional)

Remote1 ▼

Name this Remote*

Sprinklers - Remote 1

When a remote is published onto the dashboard it will automatically refresh when any of the values on the table change.

Event Timeout

You can adjust the refresh frequency by changing the **Event Timeout** period to something that matches your personal preferences. The default value is 2 seconds which means that after 2 seconds without any changes the table on the dashboard will refresh automatically. If you wish you can set this value to never and use the refresh icon update the table.

The Event Timeout setting has no bearing on an Endpoint, and these must be refreshed manually. If the Endpoint is used to invoke changes it will already be in the correct state so a refresh is generally only need to check whether an external change has occurred.

Remote Builder Version 3.0

It's a Wrap

Well, if you made it this far you are ready to build your own remotes and get the most out of them. I look forward to seeing some of the designs that people come up with and share on the community forums.

This is the day the Lord has made, let us rejoice and be glad.