

SAVANT KNOWLEDGE

< [BACK TO ARTICLES](#)

Calling a Pro App User Scene with a Custom Workflow via SCLI Bridge

RacePoint Blueprint Programming Guide

Document Date:	August 2019 (updated)
Document Supports:	da Vinci 8.3 and higher

This document provides guidelines for configuring a custom workflow within RacePoint Blueprint to call a Pro App User Scene using the Savant Command Line Interface (SCLI) Bridge utility. The resulting workflow can then be added to a state trigger to be run under any desired conditions, for example:


- When a keypad button is pressed
- When a given service or zone becomes active
- On a schedule with added conditional rules not available within the Pro App Scene scheduler

For further information on configuration of state triggers or custom workflows, refer to the relevant articles below:

State Trigger Overview: [State Trigger Programming Guide \(/Customers/apex/cx_knowledge2#!/articles?url=State-Trigger-Overview-State-Trigger-Programming-Guide-1423469857424\)](#)

Custom Workflow Development: [RacePoint Blueprint Programming Guide \(/Customers/apex/cx_knowledge2#!/articles?url=Custom-Workflow-Development-RacePoint-Blueprint-Programming-Guide-1423469833047\)](#)

The following sections will describe how to connect to the system Host and retrieve the needed Scene information, and add a custom workflow using a Run Shell Script action in Automator to call the Scene to a RacePoint Blueprint configuration.



IMPORTANT NOTE!
When running any of the terminal command line actions described in this article with a Mac-based Host (Savant Pro Host, Savant Super Pro Host), the full filepath for the sclibrige utility must be added, as shown below:


Smart Host/S2/Embedded Host devices:

```
sclibrige getSceneNames
```

OS X/Mac/Pro Hosts:

```
/Users/rpm/Applications/RacePointMedia/sclibrige getSceneNames
```

This applies to every command sent to sclibrige on an OS X Host.



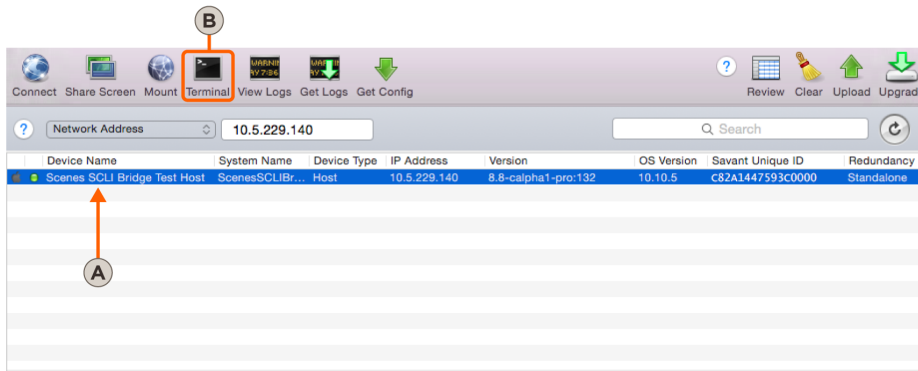
Helpful Information:
SCLI Bridge has a help menu available from the command line. To access this menu in a terminal connected to the Host, run the command:

```
sclibrige -h
```

Retrieving Scene Data from the Host via Terminal

In order to proceed with collecting the Scene data needed to configure the workflow, an SDE with access to the Host via System Monitor is required. Follow the steps below to collect the needed information:

- Open System Monitor via SAM and click once to highlight the Host in the Scanner window [A]
- Click the **Terminal** icon from the toolbar menu to open an ssh session with the Host [B]



- When prompted, enter the password for ssh access (RPM by default)
- Enter the following command:

```
sclibridge getSceneNames
```

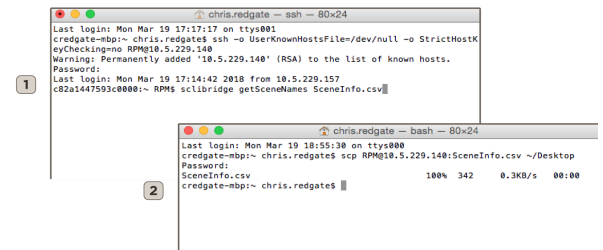
```
c82a1447593c0000:~ RPM$ sclibridge getSceneNames
Lights Out,E2501ED3-7A65-4EBF-B041-56D7A37C1C7,00A0B3E70E340060
Party,7E5B7126-6F1B-4777-9EAB-FBA644385E82,ender0937@gmail.com
Chill,1DC82316-AFCD-409F-B045-D48227C5FAE3,ender0937@gmail.com
Let There Be Light,221E5473-4F73-4923-93B4-84F36A6A1C0,00A0B3E70E340060
c82a1447593c0000:~ RPM$
```

This will print a list of all configured Scenes in the format:
[Scene Name],[Scene ID],[user email address]

- Note all of the relevant information for the Scene(s) to be run using a custom workflow.

(Optional)

- To copy the Scene information into a .csv file, run the command:
sclibridge getSceneNames [file name].csv
Any chosen filename can be used. This will copy the data to a .csv file with the entered name located in the Host's root folder.
- The .csv file can then be copied to the SDE by mounting the Host drive (Pro Host only,) via System Monitor, or by opening a new terminal window on the SDE and copying the file using the scp command as follows:
scp RPM@[Host IP Address]:[.csv filename] ~/[Directory on SDE to copy to]
For example:
scp RPM@10.5.229.140:SceneInfo.csv ~/Desktop
- The .csv file can be viewed to refer back to Scene information at a later time as needed.
- Note all relevant information for the Scene(s) to be run using a custom workflow.



Retrieve Scene Info: Alternate Method:

Using some pre-8.10.1 versions of the da Vinci runtime software, Savant Support has encountered cases in which the getSceneNames command fails to print the expected information. If this behavior is encountered in the field, the alternate method described below can be used to collect the information:


- Leave the Terminal window with open ssh session to the Host in which the getSceneNames command failed open, it will be used in later step.
- Open System Monitor and connect to the Host by double-clicking it in the Scanner window.
- Select the **Processes** tab from the options menu on the left.
- In the list of Host processes, locate the **DIS-dashboard** process and click once to highlight it.
- Click under the **Log Level** column, then use the dropdown menu to set the level to **Debug**.

Process ID	Process Name	Status	Version	Build	Log Level	% C...	# Threads
644	activity	Running	8.8-calpha1-pro	132	Info	0.06	29
619	API State Consumer Da...	Running	8.8-calpha1-pro	132	Warning	0.10	10
604	avc-Pro Remote	Running	8.8-calpha1-pro	132	Warning	0.10	3
603	avc-Scenes SCLI Bridge...	Running	8.8-calpha1-pro	132	Warning	0.30	36
622	CameraServer	Running	8.8-calpha1-pro	132	Info	0.06	17
598	Debug and Managemen...	Running	8.8-calpha1-pro	132	Error	0.50	34
609	DIS-channelFavorites	Running	8.8-calpha1-pro	132	Warning	0.10	12
615	DIS-dashboard	Running	8.8-calpha1-pro	132	Debug	0.40	19
614	DIS-equalizer	Running	8.8-calpha1-pro	132	Warning	0.10	13
616	DIS-serviceCall	Running	8.8-calpha1-pro	132	Warning	0.10	11
613	DIS-sleepTimer	Running	8.8-calpha1-pro	132	Warning	0.30	11
610	DIS-userData	Running	8.8-calpha1-pro	132	Error	0.10	12
602	DISsupport	Running	8.8-calpha1-pro	132	Warning	0.10	9
621	edm	Running	8.8-calpha1-pro	132	Warning	0.84	170
600	Media Controller Interface	Running	8.8-calpha1-pro	132	Notice	0.00	14
597	Monitor Daemon	Running	8.8-calpha1-pro	132	Error	2.50	18
645	PeripheralDeviceManager	Running	8.8-calpha1-pro	132	Warning	0.20	17
607	qcManagerd	Running	8.8-calpha1-pro	132	Warning	0.70	7
617	rpmAPIHost	Running	8.8-calpha1-pro	132	Warning	0.00	10

- Return to the System Monitor Scanner window and select the Host by clicking once.
- Open a terminal running a tail of Host process logs by selecting the **View Logs** option in the top toolbar menu, then enter the password for ssh access when prompted.
- Return to the initial terminal window with active ssh session with the Host, where getSceneNames failed (or open a new one by selecting the **Terminal** option in the scanner window).
- Position the two terminal windows so that both can be viewed at the same time, then run the sclibridge getSceneNames command again. The Scene Data will be visible within the Host live logs.
- Highlight, copy, and paste the Scene data into a text file.

User-added image

Note: Live Logs will continue auto-scrolling as Host actions occur, but can be fixed in place by scrolling up/down within the terminal window.



Helpful Information: Note that any Global Scenes will show an alphanumeric string in place of the user email address. This should be used in the activateScene action described in the steps below in the [User Email] parameter.

Test the Scene using SCLI Bridge via Terminal

To make sure the Scene data gathered is correct and will run the Scene as expected, test the command in an ssh session with the Host via Terminal as described below:

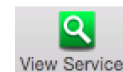
- Return to the terminal window with an active ssh connection to the Host, or open a new one via System Monitor.
- To run the selected Scene, the command format is as follows: `sclibridge activateScene '[Scene Name]' '[Scene ID #]' '[User Email]'`
Each parameter after the activateScene command must be enclosed in single quotes/apostrophes so that it will be read as one continuous string. For example:
`sclibridge activateScene 'Party' '2D374011-867D-426A-8AA3-228E3896C3EE' 'user123@gmail.com'`
- Confirm that this has activated the Scene as expected.

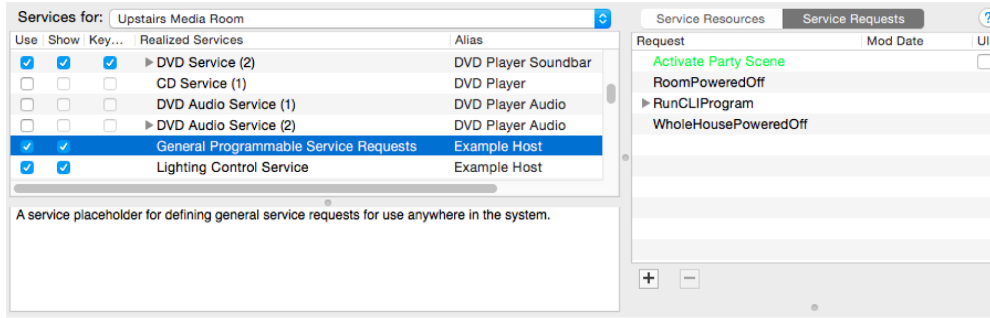
```
c82a1447593c0000:~ RPMS sclibridge activateScene 'Party' '7E5B7126-6F1B-4777-9EA8-FBA644385EB2' 'enduser0937@gmail.com'
c82a1447593c0000:~ RPMS █
```

Adding the activateScene Action to a Custom Workflow

Once the activateScene action has been tested and confirmed to function properly, a custom workflow can be configured to run the command using the **Run Shell Script** action within Automator.

- Within the RacePoint Blueprint configuration for the system, navigate to the **View Services** window by selecting the icon from the toolbar menu, or by choosing Services for [config name] under the **View** menu.
- Choose the Zone where the custom workflow will be created from the dropdown, then select **General Programmable Service Requests** from the list of services.
- Add a new workflow to the **Service Requests** list by clicking the + icon, then enter a name for the workflow. Once added it should appear in the Service Requests frame in green.





- Double-click the new workflow to open it in **Automator**.
- From the Library dropdown in the far left frame, select **Utilities**, then scroll down, or use the search field, to locate the **Run Shell Script** action.
- Click and drag the action into the workflow.
- The **Shell** dropdown should be set to `/bin/bash`, which is the default.
- The **Pass Input** dropdown must be set to "as arguments".
- The text of the script will differ depending on the Host type, (due to different filepaths for the directory containing SCLI Bridge,) but generally follows the same format used to test Scene activation via terminal session described above.

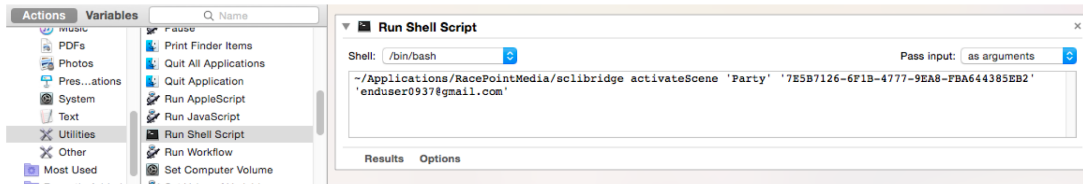
For all **Pro Hosts**, use the format:

```
~/Applications/RacePointMedia/sclibridge activateScene '[Scene Name]' '[Scene ID]' '[User Email]'
```

For all **Smart Hosts**, there is no need to specify the directory, use the format:

```
sclibridge activateScene '[Scene Name]' '[Scene ID]' '[User Email]'
```

- Before exiting the Automator window, **Save** the workflow manually, (File > Save).



Helpful Information:

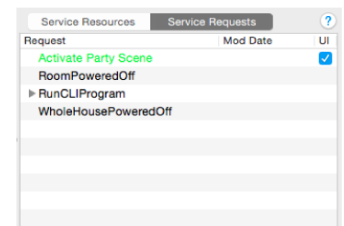
- As noted above, for Global Scenes not tied to a single User, the `getSceneNames` command will print an alphanumeric string in place of the User email. Use this string in exactly the same way within the shell script, in place of the User email.
- All argument parameters following **activateScene** should be enclosed by single quotes (') and separated by a single space.

Test the Custom Workflow

(Optional)

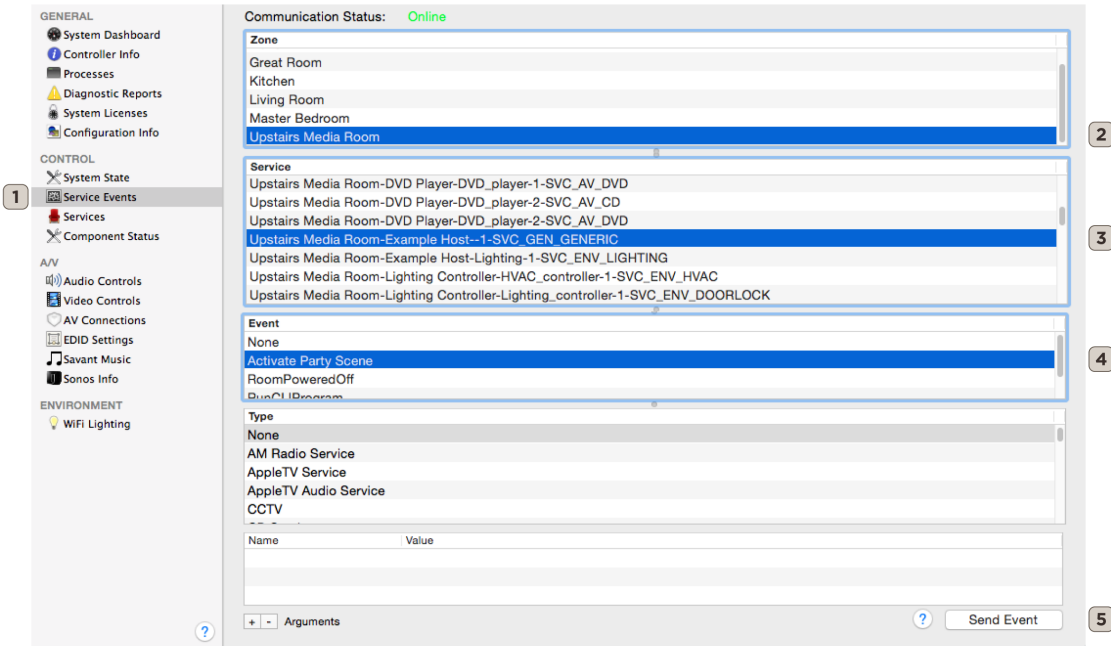
Once the custom workflow with the Run Shell Script action to call a Scene has been added to the RacePoint Blueprint configuration, it can optionally be tested, and then added to a state trigger, a Button Remote button, or used in any other way.

- In the Service Selector window where the workflow was created, click to enable the checkbox in the UI column to the right of the Service Requests frame. This will cause the workflow to appear within the Savant Pro App UI under the Commands service type on the Home page and in the configured Zone.
- Clear the current configuration, then save and upload the new version and test running the workflow via the Pro 8 App to confirm that the Scene is called as expected.



Note: Workflows can also be tested via System Monitor:

- Connect to the Host from the Scanner window of System Monitor by double-clicking,
- Select the **Service Events** tab.
- Select the Zone where the workflow was added to the configuration,
- Choose General Programmable Service Requests (SVC_GEN_GENERIC) from the Services options
- Select the desired custom workflow from the list. The Event and Type fields can be left with no selection in this case.
- Click the **Send Event** button at the lower right to run the workflow.



Adding the Workflow to a State Trigger

Once the workflow to call a Scene has been successfully configured, for most common use cases it will need to be added to a state trigger to run under set conditions. The images below show two variations on a common application; running the Scene workflow based on a (3rd-party) keypad button press.

This is just one of many potential use cases. The workflow can be added to a trigger to run under any desired state-based conditions. For further information on state trigger configuration, refer to: [State Trigger Overview: State Trigger Programming Guide \(/Customers/apex/cx_knowledge2#!/articles?url=State-Trigger-Overview-State-Trigger-Programming-Guide-1423469857424\)](#)

Important Note:

The example triggers below should not be copied directly, as the specific conditional states used, their arguments/parameters, and the target values will all vary based on a number of different factors. All states used in any trigger should first be monitored to confirm they respond as expected to desired trigger conditions.

Calling a Scene when Button is Pressed

The state trigger pictured below will simply call the selected Scene workflow when the **CurrentButtonStatus** state for the target keypad button changes to "Press".

Triggers

Enable	Group/Trigger
<input checked="" type="checkbox"/>	Party Scene Button

Description

When any of these states change value, evaluate the Rules

State Name: CurrentButtonStatus_1... State Scope: Lighting Controller.Lighting_controller On Set:

Match the following rule

if (rules)

State Name	State Scope	Data Type	Test Condition	Value	Offset
CurrentButtonStatus_140_3	Lighting Controller.Light...	string	is equal	Press	0
DeviceID	Lutron Device ID			140	
ButtonNumber	'1' - '999'			3	

and or not (->)<-

Run the following actions: Once

then

Request	Service	Arguments	Value	After De...	Override
Party Scene	Upstairs Media Room-Example Host--1-General Program...		0		

else

Request/State/Scene: Service/Scope: Arguments/Data Type: Value: After De...: Override:

Drag a state or request here for a "else" action

Programming View Export Selection... Import... Cancel Save

Toggling Two Scenes Based on LED Status

The second example trigger in the image below shows a configuration in which one of two Scenes is called based on the **IsCurrentLEDOn** state, for toggle buttons with LEDs that track their status.

- The Scene called by the workflow added to the **Then** field will be activated when the LED comes on.
- The Scene called by the workflow added to the **Else** field will be activated when it turns off.

User-added image

Copyright ©2018 Savant Systems LLC (<https://www.savant.com>) All Rights Reserved