Project Editor and BitWise Touch

Module API

API version 1.9

5/16/2013

MODULE API REFERENCE V1.9

TABLE OF CONTENTS

Document Version History

Version	Date	Author	Comments
1.0	11/24/12	M. Buster	Original Document
1.1	1/23/13	S. Miller	Added new API commands
1.2	5/16/13	S. Miller	Added cover page, UPnP methods and other misc. 1.9 API enhancements

INTRODUCTION, AUDIENCE, & CONFIDENTIALITY

INTRODUCTION:

This document is intended to describe the various JavaScript API functions and callbacks that are available to interact with the BitWise Controls GUI Elements and Controllers.

Module API v1.8 introduces a number of new capabilities, and also provides a completely new API interface which has been restructured and consolidated to be easier to use, maintain, and extend.

Any API functions not specifically included in this document should be considered to be deprecated. This means that they may be removed (and thus cease to work) at any time in a future release. As of initial release, no prior or existing API functionality has been removed, but developers should begin taking steps to implement the new Module API functionality to ensure compatibility moving forward.

AUDIENCE:

This document is intended for use by developers who wish to create two-way Script Devices for the BitWise Controls platform. **Developers must have a working knowledge of basic JavaScript programming fundamentals, as well as familiarity with the BitWise Controls platform in general.**

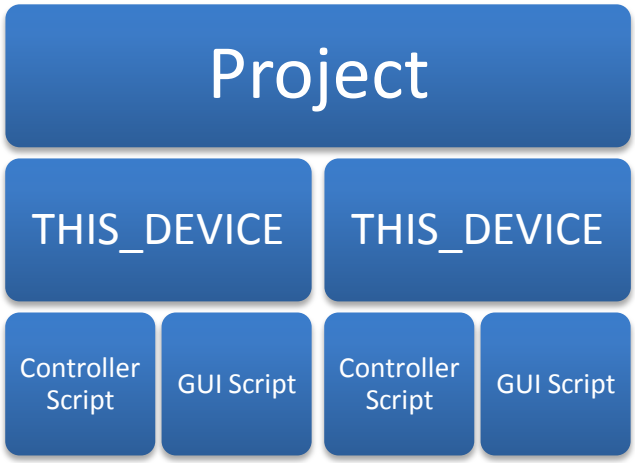
CONFIDENTIALITY:

API REFERENCE

The following tables detail API objects and methods.

The API is broken down into three major objects

- **THIS_DEVICE** – The main object for the script device. This object contains properties and callback events relevant to the specific script device.
- **GUIAPI** – The main object for API methods available on the GUI (within BitWise Touch)
- **CTRLAPI** – The main object for API methods available on the controller
  (BC-1 or BC2 only, BC-4 does not contain or run any JavaScript)



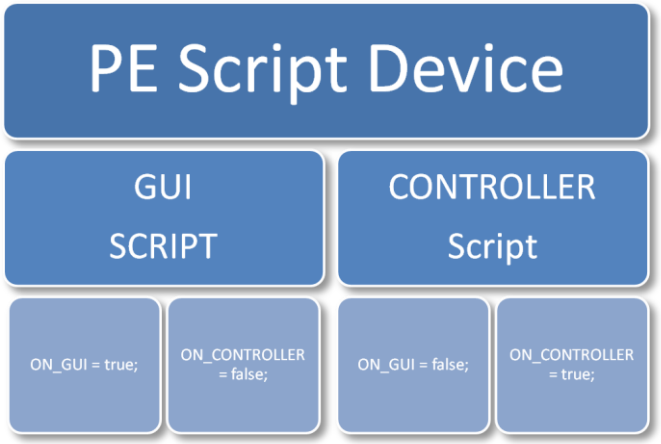
ENVIRONMENT VARIABLES

At runtime, each Script Device is loaded on the GUI ('app' side) and on the BC1 and BC2 Controllers. Some Module API functions may be available only when executing on a GUI, on a Controller, or both. For example, Controllers do not implement any DOM-related functionality, whereas when running on a GUI, full access to the DOM is provided. The following environment variables can be used to selectively execute code based on where it is running.

Name	ON_GUI	ON_CONTROLLER	Type	Description
ON_GUI	true	false	Boolean	Together with the ON_CONTROLLER environment variable, ON_GUI allows the conditional execution (or declaration) of code based on where it is being run.
ON_CONTROLLER	false	true	Boolean	Together with the ON_GUI environment variable, ON_CONTROLLER allows the conditional execution (or declaration) of code based on where it is being run.

ENVIRONMENT VARIABLES EXAMPLE:

```
if (ON_GUI){
        //Do something GUI-specific
}else if (ON_CONTROLLER){
      //Do something Controller-specific
}
```



SCRIPT DEVICE OBJECT & PROPERTIES

The developer may read various properties of the Script Device at runtime. The functions in the table below are available both ON_GUI and ON_CONTROLLER.

Object	Property	Type	Description
THIS_DEVICE		Object	THIS_DEVICE is the object wrapper, which contains all functions, callbacks, and properties of a Script Device.
	.ParentControllerType	Number	Will be 1, 2, or 4 depending on which controller contains the Script Device
	.Name	String	The name of the Script Device
	.Port	Number	The Port setting of the Script Device
	.IP	String	The IP address of the Script Device (if it is not an RS232 device)
	.SETTINGS	Object	The SETTINGS object provides a reference to anything defined in the 'User Settings' tab of the Script Device properties window.

SCRIPT DEVICE PROPERTIES EXAMPLE:

```
var myName = THIS_DEVICE.Name;
var myIP = THIS_DEVICE.IP;
var myPort = THIS_DEVICE.Port;
var myController = THIS_DEVICE.ParentControllerType;
if (ON_GUI){
    alert(myName + "@" + myIP + ":" + myPort + " on BC" + myController);
}
```

See Also: Example Project – *SettingsExample.bwe*

SCRIPT DEVICE CALLBACKS

Developers may implement the following callback functions in the "Shared Script" section of the Script Device. These callbacks can be used to trigger various tasks, such as querying for updated device status. **Starting with PE version 1.8, these callbacks will automatically be added when you create a new Script Device.**

Object	Property	Parameters	Description
When "ON_CONTROLLER == true" ONLY!			
THIS_DEVICE	**.OnTCPClientConnected**	N/A	Will be called when the BC1 connects to the TCP Client Device.
	.OnTCPClientClosed	N/A	Will be called when the TCP Client connection is closed for this device.

Object	Property	Parameters	Description
When "ON_GUI == true" ONLY!			
THIS_DEVICE	**.OnGUILoaded**	Page (String)	Will be called with the name of the page that was loaded. Use to perform any initialization your module may require.
	.OnGUIWake	N/A	Called when the iOS or Android device wakes from sleep.
	.OnGUIWillGoto	Page (String)	Called with the name of the page the GUI is about to navigate to.
	.OnPopupOpening	Name (String)	Called when a popup is opening.
	.OnPopupOpened	Name (String)	Called when a popup is presented.
	.OnPopupClosing	Name (String)	Called when a popup is closing.
	.OnPopupClosed	Name (String)	Called when a popup is closed.
	.OnUPNPDiscovery	Data (String)	Callback for when an UPnP device is discovered.

SCRIPT DEVICE CALLBACKS EXAMPLE:

This example shows an implementation of each Script Device callback, as would appear in the Shared Script section of the Script Device.

```
/*****Default Script Device Callbacks ****/

THIS_DEVICE.OnGUILoaded = function(Name){
	//This function is called by the API whenever a new page is loaded
	try{

	}catch(e){
		alert(THIS_DEVICE.Name + ".OnGUILoaded error:" + e);
	}
};


THIS_DEVICE.OnGUIWake = function(){
	//This function is called by the API whenever the iOS or Android device wakes from sleep
	try{

	}catch(e){
		alert(THIS_DEVICE.Name + ".OnGUIWake error:" + e);
	}
};

THIS_DEVICE.OnGUIWillGoto = function(Name){
	//This function is called by the API before a page jump
	try{

	}catch(e){
		alert(THIS_DEVICE.Name + ".OnGUIWillGoto error:" + e);
	}
};
```

See Also:

Example Project – *PopupExample.bwe*

Example Project – *RokuExample.bwe*

GUIAPI OBJECT & VERSION

The GUIAPI Object provides many useful mechanisms for interacting with GUI Elements, displaying Feedback, etc…

Wrapper	Property Name	Type	Description
GUIAPI		Object	GUIAPI is the wrapper object containing all GUI API functions. The GUIAPI object is a 'window' or 'global' level object, and is already instantiated by the time any user code is executed, thus no constructor is provided or necessary to the developer.
	.GetVersion	String	Can be used to check current GUIAPI version being used
	.VersionMeetsMinimum	Boolean	Pass this function a float to see if the current GUIAPI.Version exceeds or is equal to the supplied value

GUIAPI VERSION EXAMPLE:

```
 
    THIS_DEVICE.OnGUILoaded = function (Name) {
        //This function is called by the API whenever a new page is loaded
        try {
            var apiMinimum = 1.8;
            if (typeof GUIAPI != "undefined") {
                var apiVersion = GUIAPI.GetVersion();
                if (GUIAPI.VersionMeetsMinimum(apiMinimum)) {
                    //All is well, do stuff
                    return;
                }
            }
            alert (THIS_DEVICE.Name + " Requires GUIAPI Version " +
apiMinimum.toString() + " or higher");
        } catch (e) {
            alert (THIS_DEVICE.Name + " OnGUILoaded Error: " + e);
        }
    }; 
```

GUIAPI POPUPCONTROLLER OBJECT, FUNCTIONS, & CONSTANTS

The following objects and constants allow developers to show and hide Popup pages via JavaScript.

Note: At runtime, only Popup pages referenced by button navigation on the current page will be accessible.

Wrapper	Property Name		Type	Parameters	Description
GUIAPI	**.PopupController**		Object		This object provides methods to show and hide Popup pages
	.PopupController	**.ShowPopup**	Method	Name (String), From (String), To (String), Speed (String), Modal (Boolean), SelectOnly (Boolean)	Name – The name of the Popup page to show From – The starting position To – The ending position Speed – The animation speed to use Modal – if true, block the rest of the page SelectOnly – If true, close after the first button action
		.ClosePopup		Name (String)	Name – The Popup to close
GUIAPI	**.PopupController**	**.ClosePopupThenGoto**		Name (String), Goto (String)	Name – The name of the Popup to close Goto – The name of the page to go to when the Popup close animation finishes

	.POPUPSETTINGS		Object		This object contains some useful constant values that can be used with GUIAPI.PopupController
	.POPUPSETTINGS.SPEED		Object		Contains constant values for Popup display speeds
	.POPUPSETTINGS.SPEED	**.SLOW**	String		Slow display speed
		.NORMAL			Normal display speed
		.FAST			Fast display speed
	.POPUPSETTINGS.TO		Object		Contains constants related to Popup display location
	.POPUPSETTINGS.TO	**.TOPLEFT**	String		Top Left display
		.TOPCENTER			Top Center display
		.TOPRIGHT			Top Right display
		.LEFT			Left display
		.CENTER			Center display
		.RIGHT			Right display
	.POPUPSETTINGS.TO	**.BOTTOMLEFT**			Bottom Left display
		.BOTTOMCENTER			Bottom Center display
		.BOTTOMRIGHT			Bottom Right display
	.POPUPSETTINGS.FROM		Object		Contains constants related to popup start position

	.POPUPSETTINGS .FROM	**.FADE**	String		Fade in/out
		.LEFT			Slide In/Out from left
		.RIGHT			Slide In/Out from right
		.TOP			Slide In/Out from top
		.BOTTOM			Slide In/Out from bottom
	.POPUPSETTINGS.MODAL		Boolean		Show Popup modally (block the rest of the page)
	.POPUPSETTINGS.NONMODAL		Boolean		Show Popup without blocking the page
	.POPUPSETTINGS.SELECTION		Boolean		Close Popup after any item is pressed
GUIAPI	**.POPUPSETTINGS.CONTROL**		Boolean		Allow multiple items to be pressed on Popup

POPUPCONTROLLER EXAMPLE:

```
var name = "POPUP_Volume";
var from = GUIAPI.POPUPSETTINGS.FROM.FADE;
var to = GUIAPI.POPUPSETTINGS.TO.BOTTOMCENTER;
var speed = GUIAPI.POPUPSETTINGS.SPEED.NORMAL;
var modal = GUIAPI.POPUPSETTINGS.NONMODAL;
var style = GUIAPI.POPUPSETTINGS.CONTROL;
GUIAPI.PopupController.ShowPopup(name, from, to, speed, modal, style);
```

See Also:

Example Project – *PopupExample.bwe*

GUIAPI SCROLLING LIST OBJECTS, PROPERTIES & METHODS

The Scrolling List GUI element allows the user to browse and select dynamically presented content such as media lists or menu items. Scrolling Lists can be added to a GUI page via the GUI Editor, but this only creates the visual, or 'front-end' aspect of the list. The developer must provide display and selection handling code.

Wrapper	Property Name		Type	Parameters	Description
GUIAPI	**.GetListController**		Object	Tag (String)	Get a reference to the ListController instance linked to the provided Tag. Tag correlates to the 'Custom Tag' value of the Scrolling List in the GUI Editor.
	.ListController		Object		The ListController object provides all the necessary methods and properties to display and interact with a Scrolling List GUI Element.
	.ListController	**.Items**	Array		An array, which contains any list items to be displayed. All standard JavaScript array manipulation methods apply, such as push, pop, splice, etc.
		.Clear	Method	None	Call to remove all list items
		.Refresh			Must be called to refresh appearance after any change to the Items array
		.SetSelectionHandler		Callback (Function)	Pass in a function to be called whenever a List Item is selected

		.SetLongPressHandler		Callback (Function)	Pass in a function to be called when a List Item is pressed and held
		.ShowLoading		Loading (Boolean)	Set true to show loading animation
	.ListItem		Object	Text (String), Selectable (Boolean)	Create a new ListItem instance with the provided Text. If Selectable is false, the ListItem will not trigger selection or long press handlers
GUIAPI	**.ListItem**	**.SetText**	Method	Text (String)	Change the display text
		.GetText			Return the display text
		. SetThumbnailImageSrc			Source-The network path to an image file to be displayed as a thumbnail on the List Item

LISTCONTROLLER EXAMPLE:

```
 
THIS_DEVICE.ListTag = "MY_CUSTOM_TAG";
if (GUIAPI.PageHasTaggedList(THIS_DEVICE.ListTag)) {
    //grab a reference to the Scrolling List GUI Element's ListController
object
    var listController = GUIAPI.GetListController(THIS_DEVICE.ListTag);
    //Clear it and show the loading animation
    listController.Clear();
    listController.ShowLoading(true);
    //create generic ListItems to display
    for (var i = 0; i < num; i++) {
        var itemText = "Item " + i.toString();
        var selectable = true;
        var item = new GUIAPI.ListItem(itemText, selectable);
        listController.Items.push(item);
    }
    listController.SetSelectionHandler(
        function (Item) {
            GUIAPI.ShowFeedback("FBTEST", "You Selected " +
Item.GetText(), true);
        }
    );
    listController.SetLongPressHandler(
        function (Item) {
            GUIAPI.ShowFeedback("FBTEST", "You Long-Pressed " +
Item.GetText(), true);
        }
    );
    listController.Refresh();
}
```

See Also:

Example Project - *ListExample.bwe*

Example Project - *RokuExample.bwe*

GUIAPI HTTP OBJECT & METHODS

These objects and methods provide a simple mechanism to create and handle HTTP POST and GET requests asynchronously, utilizing native functions at the OS level. You may also use the standard JavaScript XMLHTTPRequest functionality if the device you are controlling requires special consideration, such as more comprehensive POST support.

Wrapper	Property Name		Type	Parameters	Description
GUIAPI	**.HTTP**		Object		This object provides convenience functions to easily communicate with HTTP devices.
		.AddHandler	Method	Tag (String), Handler (Function)	Register a callback function as available to DoGET and DoPOST responses. The Tag string is used as the identifier for this callback.
		.RemoveHandler		Tag (String)	Make a previously registered callback function unavailable to DoGET and DoPOST responses
		.DoGET		Tag (String), URL (String)	Tag – The Tag of the previously registered function to call with the result of the HTTP GET request URL-The url (including any query string) for the GET request
		.DoPOST			Tag – The Tag of the previously registered function to call with the result of the HTTP POST request URL-The url (including any query string) for the POST request

GUIAPI	**.HTTP**	**.DoPOSTEX**		Callback (Function), URL (String), Headers (Object), Headers (String)	Extended POST. Headers is an object containing each header name and value you want to specify. Content is a string, the body of the content you want to post.

HTTP EXAMPLE

```
      var myGETHandlerTag = "MyGETHandler";
           GUIAPI.HTTP.AddHandler(myGETHandlerTag,function(response){
               try{
                           GUIAPI.Debug.WriteLine(unescape(response));
                  }catch(e){
                           alert(e);
                  }
           });
           var url =
"http://en.wikipedia.org/w/api.php?action=query&titles=hulk_hogan&format=json";
           GUIAPI.HTTP.DoGET(myGETHandlerTag,url);
```

See Also: Example Project - *RokuExample.bwe*

GUIAPI UPNP CONTROLLER

This object and the associated methods provide a simple mechanism to discover and subscribe to UPnP devices. Once a device is discovered, its description document s can be retrieved using HTTP.DoGET. The device can be controlled using HTTP.DoPOSTEX. For more information regarding the UPnP architecture, visit http://www.upnp.org

Wrapper	Property Name	Type	Parameters	Description
GUIAPI	**.UPNPController**	Object		This object provides convenience functions to easily communicate with UPNP devices.
	.DiscoverSSDP	Method	SSDP (String), Timeout (Number)	Do an UPnP discovery using the supplied SSDP and timeout.
	.AddSubscriptionCallbackHandler	Method	Tag (String), Handler (Function)	Register a callback function that gets called when a subscribed event changes. The Tag string is used as the identifier for this callback.
	.Subscribe	Method	Tag (String), Host (String), Path (String), Timeout (Number)	Sets up a native listener for event notifications from the device and sends the subscription request. Tag corresponds to a tag used to register a callback via AddSubscriptionCallbackHandler.

GUIAPI STORED OBJECT FUNCTIONS

Since JavaScript variables are not persistent across page jumps or app restarts, sometimes it is useful to store a data object for later retrieval. These methods allow you store data objects (which will be serialized and de-serialized via JSON.stringify/parse) as files by the native app backend. **Note that these files will be stored as plain text, so care should be taken that no sensitive user or module data should be stored via these methods.**

Wrapper	Property Name	Type	Parameters	Description
GUIAPI	**.StoreSerializedObject**	Method	Tag (String), Data (Object)	Serializes and stores a JavaScript object using the provided Tag value for later retrieval. Useful for making data persistent across page jumps or app sessions.
	.RetrieveSerializedObject	Method	Tag (String), Callback (Function)	Query app backend to retrieve a previously stored object, referenced by Tag. Callback will be called with the stored object if available.

STORED OBJECT EXAMPLE:

```
var obj = {
        name:"Smilin' Donnie",
        place:"Pablo's room"
};

var mySessionTag = THIS_DEVICE.Name + "_SESSION_TAG";
GUIAPI.StoreSerializedObject(mySessionTag,obj);
GUIAPI.RetrieveSerializedObject(mySessionTag,function (obj,err){
        if(obj){
                var msg = obj.name + " was last seen in " + obj.place;
                GUIAPI.Debug.WriteLine(msg);
        }else{
                GUI.Debug.WriteLine(err);
        }
});
```

GUIAPI DEBUG FUNCTIONS

This function makes it possible to monitor runtime activity of your module via Project Editor's new Debug Monitor window, available in the Tools menu.

Wrapper	Property Name	TYPE	Parameters	Description
GUIAPI	**.Debug.WriteLine**	Method	Message (String)	Broadcasts a debug message

GUIAPI DEBUG.WRITELINE EXAMPLE:

```
var myMessage = "Something important happened";
GUIAPI.Debug.WriteLine(myMessage);
```

GUIAPI UTILITY FUNCTIONS

Wrapper	Property Name	TYPE	Parameters	Description
GUIAPI	**.ShowFeedback**	Method	FBID (String), Value (String), Local (Boolean)	FBID- The Feedback ID Value- The value to be displayed Local- True to only display on Local GUI, False to propagate value throughout system
	.GetFeedback	Method	FBID (String)	FBID- The Feedback ID of the stored value to retrieve
	.PageFBIDsWithPrefix	Array	Prefix (String)	Prefix – The string to use to find prefixed FBIDs on the current page
	.PageHasTaggedList	Boolean	Tag (String)	Tag – The Custom Tag value to search for
	.PageHasTaggedImage	Boolean	Tag (String)	Tag – The Custom Tag value to search for
	.SetTaggedImageSource		Tag (String), Source (String)	Tag – The Custom Tag value to search for Source – The path to the image file
	.DefaultImageFolder	String		Get the GUI's image folder name
	.MakeASCII	Method	*Hex (String)	Converts an ASCII Hex string into an ASCII string. For example, '414243' becomes 'ABC'
	.MakeHex	Method	ASCII (String)	Converts an ASCII string of characters to ASCII Hex. For example, 'ABC' becomes '414243'

*Note: When the "Incoming Data is ASCII" box in the Incoming Data Area is not checked, incoming data will be the **ASCII representation of the Hex data**.

See Also: All Example Projects

CONTROLLER API OBJECT AND VERSION

The Controller API Object (**CTRLAPI**) provides mechanisms for interacting directly with a BC-1 or BC-2 controller. **Please note: API functions do not "run" on a BC4. You must use the GUIAPI object to interact with a BC4.**

Wrapper	Property Name	Type	Description
CTRLAPI		Object	CTRLAPI is the wrapper object containing all Controller API functions. The CTRLAPI object is a 'global' level object, and is already instantiated by the time any user code is executed, thus no constructor is provided or necessary to the developer.
	.GetVersion	String	Can be used to check current CTRLAPI version being used
	.VersionMeetsMinimum	Boolean	Pass this function a float to see if the current CTRLAPI.Version exceeds or is equal to the supplied value

See Also: Previous GUIAPI.GetVersion example.

CTRLAPI DEBUG FUNCTIONS

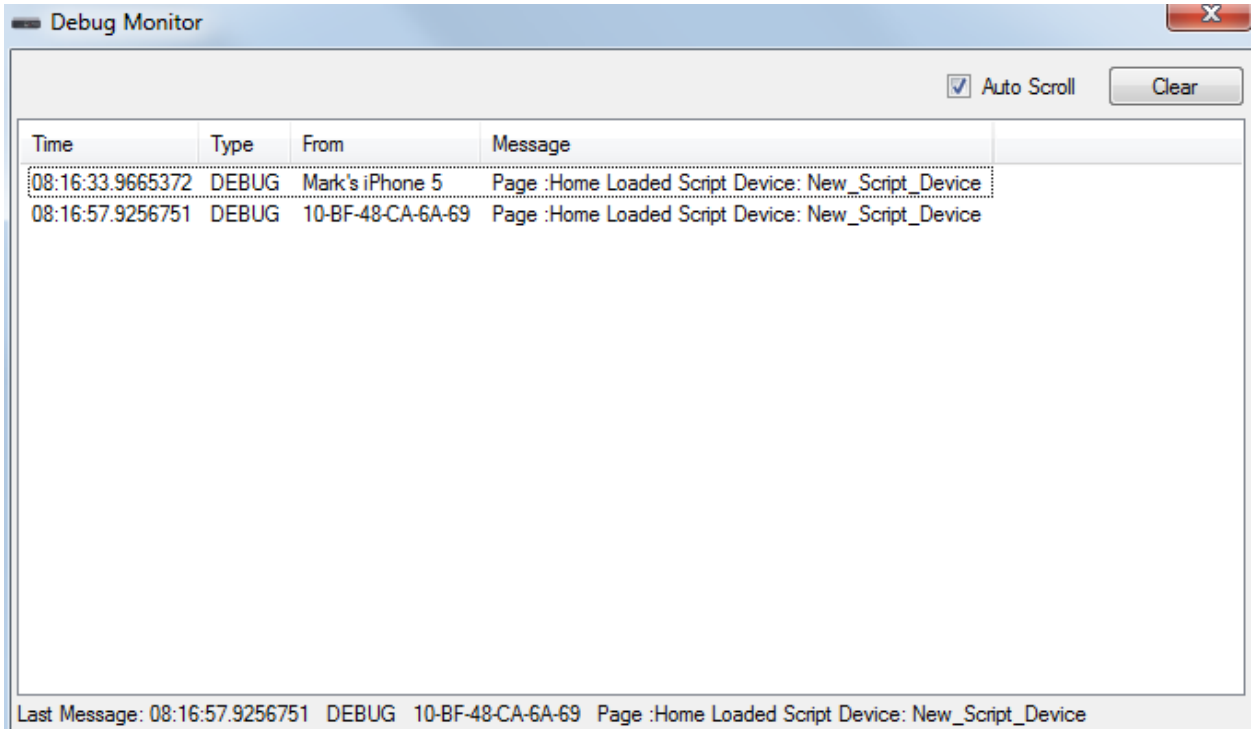
This function makes it possible to monitor runtime activity of your module via Project Editor's new Debug Monitor window, available in the Tools menu.

Wrapper	Property Name	TYPE	Parameters	Description
CTRLAPI	**.Debug.WriteLine**	Method	Message (String)	Broadcasts a debug message.

CTRLAPI DEBUG.WRITELINE EXAMPLE:

```
var myMessage = "Something important happened!";
CTRLAPI.Debug.WriteLine(myMessage);
```

Debug Monitor Window



CTRLAPI FEEDBACK UTILITIES

Use these methods to get and store data in Feedback IDs. Note that the ShowFeedback method does not have a "Local" Boolean property like the GUIAPI method does. This method will always propagate feedback to the network for any available apps to receive.

Wrapper	Property Name	TYPE	Parameters	Description
CTRLAPI	**.ShowFeedback**	Method	FBID (String), Value (String)	Store data in the specified feedback ID. FBID- The Feedback ID Value- The value to be displayed
	.GetFeedback		FBID (String)	Retrieve the current value of a feedback ID. FBID- The Feedback ID of the stored value to retrieve

```
// Alter or set feedback based on controller events

var message = "This represents a feedback string";

CTRLAPI.ShowFeedback("MY_FBID",message);
```

CTRLAPI HTTP OBJECT & METHODS

These objects and methods provide a simple mechanism to create and handle HTTP POST and GET requests asynchronously, utilizing native functions at the OS level. You may also use the standard JavaScript XMLHTTPRequest functionality if the device you are controlling requires special consideration, such as more comprehensive POST support.

Wrapper	Property Name		Type	Parameters	Description
CTRLAPI	**.HTTP**		Object		This object provides convenience functions to easily communicate with HTTP devices
		.AddHandler	Method	Tag (String), Handler (Function)	Register a callback function as available to DoGET and DoPOST responses. The Tag string is used as the identifier for this callback.
		.RemoveHandler		Tag (String)	Make a previously registered callback function unavailable to DoGET and DoPOST responses
		.DoGET		Tag (String), URL (String)	Tag – The Tag of the previously registered function to call with the result of the HTTP GET request URL-The url (including any query string) for the GET request
		.DoPOST			Tag – The Tag of the previously registered function to call with the result of the HTTP POST request URL-The url (including any query string) for the POST request

See Also: Example Project - *RokuExample.bwe*