Tutorial : Using macros

Programmable Light control

Once we can control some light with automatic turn on/off (Tutorial_Basis.html), we may want to change the on/off hours from the UI. We can use webiopi macros to achieve it. Macros are custom Pythons functions automatically bound to the REST API, so they are remotely callable. They allow to remotely trigger complex computation on the Raspberry Pi or manual events, trigger multiple GPIO at the same time, change settings...

Adding macro in the Python script

Registering macros to the WebIOPi server only requires a single Python decorator before each function. From the Framework Basis Tutorial (Tutorial_Basis.html), edit and update the Python script file, eg. **/home/pi/myproject/python/script.py**.

```
import webiopi
...
HOUR_ON = 8
HOUR_OFF = 18
...
def setup():
...
def loop():
...
def destroy():
...

@webiopi.macro
def getLightHours():
    return "%d;%d" % (HOUR_ON, HOUR_OFF)

@webiopi.macro
def setLightHours(on, off):
    global HOUR_ON, HOUR_OFF
    HOUR_ON = int(on)
    HOUR_OFF = int(off)
    return getLightHours()
```

Both get/setLightHours functions are declared to WebIOPi using the **@webiopi.macro** decorator. The first one return a string containing both ON and OFF hours separated by a semicolon. The second function casts arguments received and set ON and OFF hours as well as returning the effective hours to give a feedback to the remote UI.

Calling macros from Javascript

Using macros from Javascript is really easy. The most difficult part is to interact with the UI to retrieve data typed by the user and display the result. Edit the **index.html** file from the Framework Basis Tutorial (Tutorial_Basis.html).

First, add inputs controls and labels in the `<body>` part.

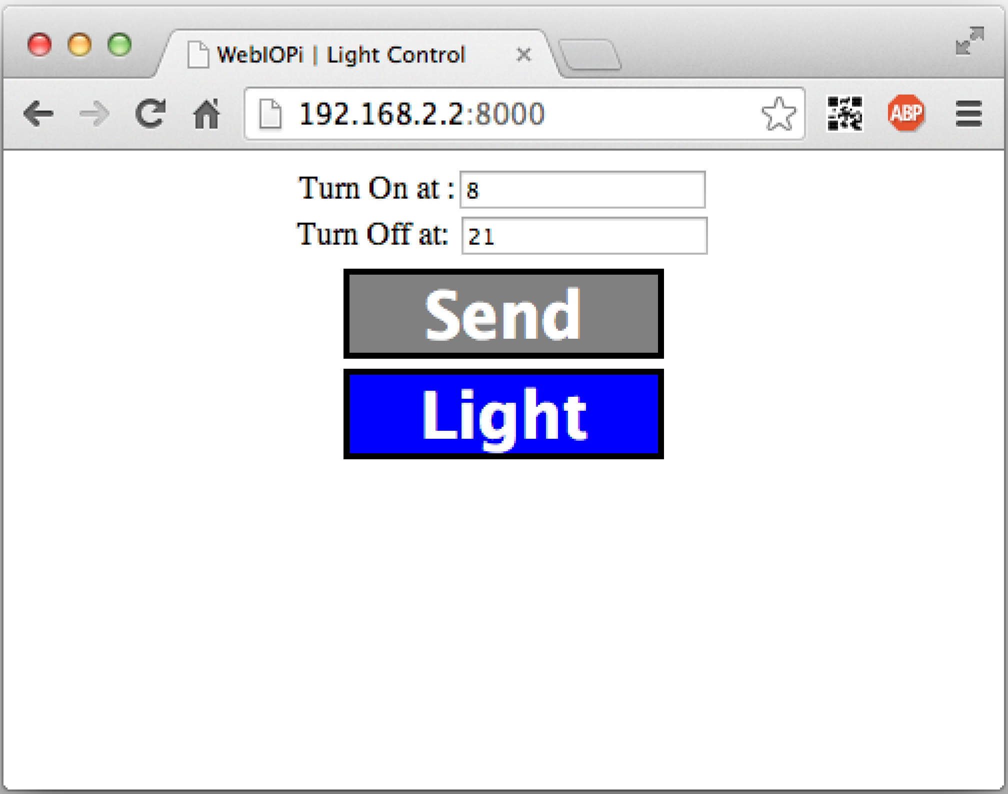
```
<html>
...
<body>
<div align="center">
Turn On at :<input type="text" id="inputOn" /><br/>
Turn Off at: <input type="text" id="inputOff" /><br/>
<div id="controls"></div>
</div>
</body>
</html>
```

Then update the **webiopi().ready()** call.

```
        webiopi().ready(function() {
        // Following function will process data received from set/getLightHours macro.
        var updateLightHours = function(macro, args, response) {
                    var hours = response.split(";");
                    // Following lines use jQuery functions
                    $("#inputOn").val(hours[0]);
                    $("#inputOff").val(hours[1]);
        }

        // Immediately call getLightHours macro to update the UI with current values
        // "getLightHours" refers to macro name
                // [] is an empty array, because getLightHours macro does not take any argument
                // updateLightHours is the callback function, defined above
                webiopi().callMacro("getLightHours", [], updateLightHours);

        // Create a button to call setLightHours macro
        var sendButton = webiopi().createButton("sendButton", "Send", function() {
           // Arguments sent to the macro
                    var hours = [$("#inputOn").val(), $("#inputOff").val()];
            // Call the macro
                    webiopi().callMacro("setLightHours", hours, updateLightHours);
                });

        // Append the button to the controls box using a jQuery function
                $("#controls").append(sendButton);

        // Create a "Light" labeled button for GPIO 17
        var button = webiopi().createGPIOButton(17, "Light");

        // Append the button to the controls box
        $("#controls").append(button);

        // Refresh GPIO buttons
        // pass true to refresh repeatedly of false to refresh once
        webiopi().refreshGPIO(true);

        });
```

Testing

As you updated the script file, you need to restart WebIOPi, then point your browser to it.



Turn On at : 8
Turn Off at: 21