Web Dashboards for Single-Board Computer Projects

Eyal Shahar

Maker Faire Bay Area 2018

@Person_es www.eyalshahar.com https://github.com/persones/shpritz

Housekeeping

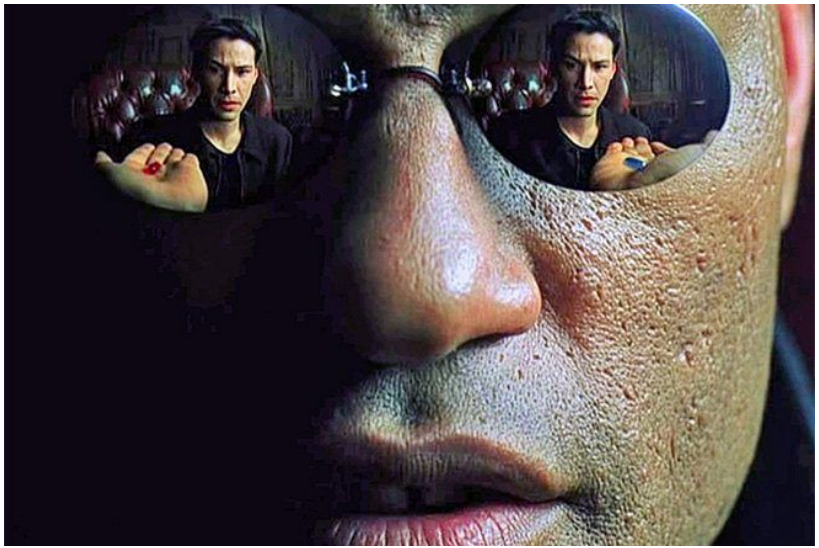
We'll make a webpage to control a R-Pi project.

We'll cover:

- Websockets
- Why Javascript is good for you
- Websockets vs. HTTP requests

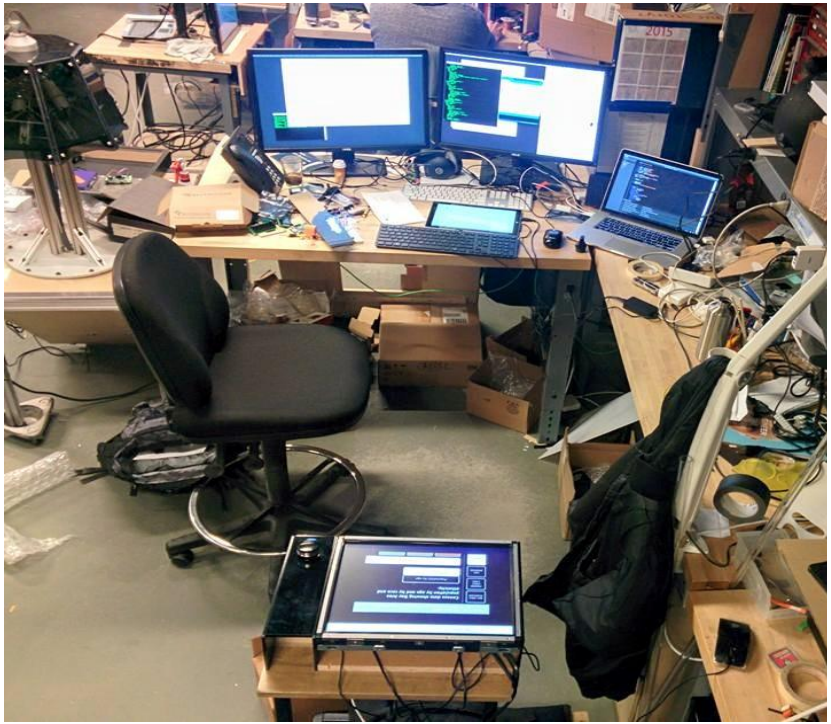
Level: intermediate

- For beginners: motivation
- For the advanced: practices



Hi!

- I’m Eyal (Pronounce like you’re reading Spanish).
- A little bit of accent. You’ll get used to it.
- Background in music. And tech. And music-tech.
- Developing new-media at the Exploratorium.



Why web dashboards?

- No need for proximity
- Potential to control from anywhere
- No physical I/O on project
- Scalable

Define the problem:

Jackpot and LeftEye

Battery Operated Trigger Sprayer

Mist-Pour

5 customer reviews

Price: $8.99 & **FREE Shipping**

Get $20 off instantly: Pay $0.00 upon approval for the Amazon.com Store Card.

In Stock.

Get it as soon as May 17 - 22 when you choose **Standard Shipping** at checkout.
Ships from and sold by COOLFLOAT.

- Battery Operated Trigger Sprayer
- Runs on 2 AA Batteries Included
- 28/400 Thread
- Comes with 11 1/4 inch Length dip tube just cut to size needed for your Bottle
- Trendy Clear Body to see all the gears running

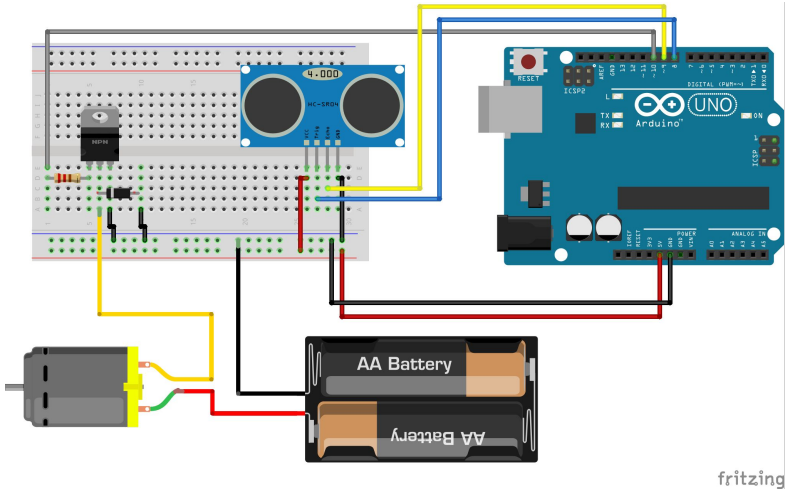
Compare with similar items

New (1) from $8.99 & FREE shipping.

Report incorrect product information.

prime student College student? Get FREE shipping and exclusive deals

4-Gallon Electric Backpack Sprayer With Stee
DJ160



JS is good for you!

- A must for web pages
- Backend with NodeJS
- Desktop with ElectronJS
- Time sensitive? On a Pi?!

```
let minDist = 30;
let trig = false;

let SerialPort = require('serialport');
let arduino = new SerialPort('/dev/ttyACM0', {
  baudRate: 9600
});

arduino.open((err) => {
  return console.log(err.message);
});

arduino.on('data', (data) => {
  // data value is distance in cm
  if (!trig) {
    let dist = data[data.length - 1];
    setShpritz(dist < minDist);
  }
});

function setShpritz(state) {
  arduino.write(state ? '1' : '0');
}
```

Shpritz!

Distance: N/A cm

Shpritz: N/A

manual shpritz

```
<!DOCTYPE html>
<html>
  <header>
  </header>
  <body>
    <h1>Shpritz!</h1>
    <p>Distance: <span id="distance">N/A</span> cm</p>
    <p>Shpritz: <span id="shpritz">N/A</span></p>
    <button onmousedown="manualTrigger(true)"
            onmouseup="manualTrigger(false)"
            onmouseleave="manualTrigger(false)">
      manual shpritz
    </button>
  </body>
</html>
```

JSON

```
{
  "distance": 130,
  "shpritz": false
}
```

- JavaScript Object Notation
- Human readable
- Cross-platform
- Values, arrays, and dictionaries

Websockets

- HTTP requests are initiated only by the client.
- Websockets allow for bidirectional communication.
- Socket.io is one library. There are others.
- Express.js is still nice for handling HTTP requests and serving pages.
- Template rendering = duplicate functions.

```
let app = require('express')();
let http = require('http').Server(app);
let io = require('socket.io')(http);

app.get('/', function(req, res){
  res.sendFile(__dirname + '/dashboard.html');
});

http.listen(5000, function(){
  console.log('listening on *:5000');
});
```

Websockets

- HTTP requests are initiated only by the client.
- Websockets allow for bidirectional communication.
- Socket.io is one library. There are others.
- Express.js is still nice for handling HTTP requests and serving pages.
- Template rendering = duplicate functions.

```
arduino.on('data', (data) => {
  // data value is distance in cm
  if (!trig) {
    let dist = data[data.lastIndexOf()];
    io.emit('dashboard', {'distance': dist});
    setShpritz(dist < minDist);
  }
});

function setShpritz(state) {
  arduino.write(state ? '1' : '0');
  io.emit('dashboard', {'shpritz': state});
}
```

```
io.on('connection', function(socket) {
  socket.on('dashboard', (msg) => {
    if (msg.hasOwnProperty('man_trigger')) {
      trig = msg.man_trigger;
      setShpritz(trig);
    }
  });
});
```

Shpritz!

Distance: N/A cm

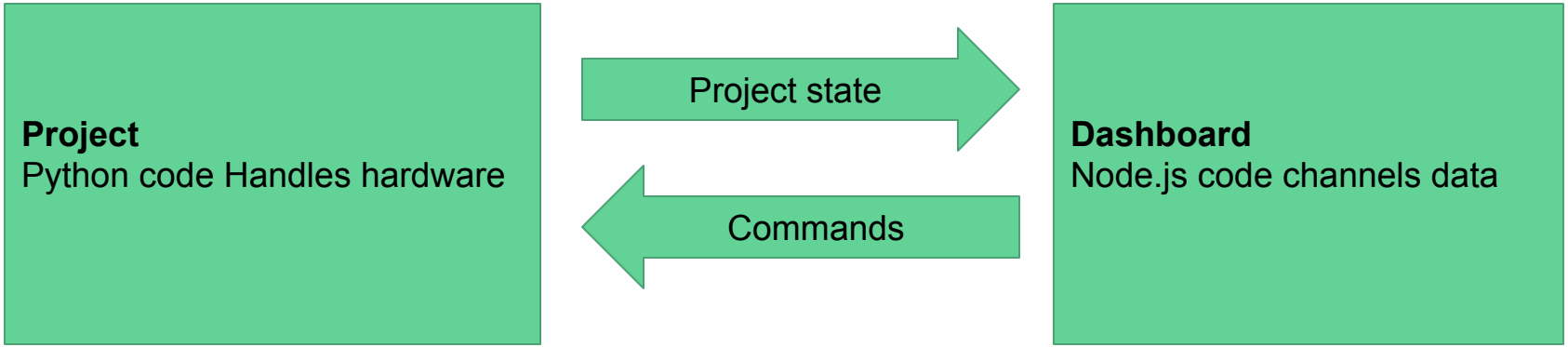
Shpritz: N/A

manual shpritz

```
<!DOCTYPE html>
<html>
  <header>
  </header>
  <body>
    <h1>Shpritz!</h1>
    <p>Distance: <span id="distance">N/A</span> cm</p>
    <p>Shpritz: <span id="shpritz">N/A</span></p>
    <button onmousedown="manualTrigger(true)"
            onmouseup="manualTrigger(false)"
            onmouseleave="manualTrigger(false)">
      manual shpritz
    </button>

    <script src="/socket.io/socket.io.js"></script>
    <script>

const socket = io.connect();
socket.on("dashboard", (msg) => {
  for (key in msg) {
    document.getElementById(key).innerHTML = msg[key];
  }
});

function manualTrigger(state) {
  socket.emit("dashboard", {man_trigger: state});
}

    </script>
  </body>
</html>
```

What about Python?

JSON over UDP



- Unlike TCP, there's no response
- Not guaranteed, but fast
- Great for inter-program communication
- Great for inter-language communication
- Great for inter-computer communication
- Perfect for JSON

Shpritz: the complicated version

```python
import socket, serial, json, threading

thresh = 30
trig = False

arduino = serial.Serial('/dev/ttyACM0', 9600)

udp = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
udp.bind(("127.0.0.1", 3001))

def udp_thread():
  global trig
  while True:
    data, addr = udp.recvfrom(1024)
    msg = json.loads(data)
    if "man_trigger" in msg:
      trig = msg["man_trigger"]
      setShpritz(trig)

def setShpritz(state):
  if state:
    arduino.write('1')
  else:
    arduino.write('0')
  message = json.dumps({"shpritz": state})
  udp.sendto(message, ("127.0.0.1", 3002))

t = threading.Thread(target=udp_thread)
t.start()

if __name__ == '__main__':
  while True:
    distance = ord(arduino.read())
    if not trig:
      setShpritz(distance < thresh)
    message = {"distance": distance}
    udp.sendto(json.dumps(message), ("127.0.0.1", 3002))
```

```javascript
let app = require('express')();
let http = require('http').Server(app);
let io = require('socket.io')(http);

app.get('/', function(req, res){
  res.sendFile(__dirname + '/dashboard.html');
});

http.listen(5000, function(){
  console.log('listening on *:5000');
});

io.on('connection', function(socket) {
  socket.on('dashboard', (msg) => {
    udpServer.send(JSON.stringify(msg), 3001, "127.0.0.1");
  });
});

const dgram = require('dgram');
const udpServer = dgram.createSocket('udp4');

udpServer.on('message', (msg, rinfo) => {
  io.emit('dashboard', (JSON.parse(msg)));
});

udpServer.on('error', (err) => {
  udpServer.close();
});

udpServer.on('listening', () => {
});

udpServer.bind(3002);
```

Setup

Connect to network:

```
# /etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
    ssid="mynetwork"
    psk="mypasswd"
}
```

Static IP address:

```
# /etc/dhcpcd.conf
interface wlan0
static ip_address=192.168.1.60/24
static routers=192.168.1.1
static domain_name_servers=208.69.40.3 208.69.40.4
```

Run on startup (from root!)

```
# crontab
@reboot /usr/local/bin/node /home/person/mf_shpritz/shpritz.js
```

Thank you

www.eyalshahar.com
eyal@eyalshahar.com
https://github.com/persones