Introduction to Node.JS

Embedded Interface Design

with **Bruce Montgomery**

Learning Objectives

Students will be able to…

- Identify the key elements of the Node.js programming environment
- Apply Node.js in development of class projects

Why use JavaScript as an IoT development tool?

- JavaScript is widely used
  - Familiarity of web developers
  - Well-documented and supported
  - Standardized with multiple implementations
- JavaScript is common in web and mobile development
  - Used by HTML5, which can be used for IoT device UIs
- JavaScript has features for embedded device programming
  - Supports asynchronous i/o and function calls
  - Event-driven hardware elements need asynchronous support
- JavaScript engines have good performance
  - Chrome V8 approaches speed of C/C++
- From Reference [1]

What is Node.js?

- Node.js is an event-driven, cross-platform, I/O server-side JavaScript environment based on V8
  - Node.js was originally developed by Ryan Dahl in 2009
  - JavaScript is normally a client-side scripting tool, embedding scripts within a webpage's HTML
- Node.js can run headless (no UI) services
- Node.js can be used to create HTTP and HTTPS web servers
- For web development, Node.js can be used to run a server-side script to dynamically change web page content before it is sent to a browser
- From Reference [1]

Why use Node.js for IoT?

- Node.js is supported in the serverless code tools of the big 3 Cloud IoT frameworks
- Strong documentation and package management system
  - Over a hundred thousand packages available via the npm package manager [2]
- Programming model well-suited to embedded devices
  - Event-driven asynchronous programming model, support for asynchronous functions
  - Lack of explicit event loop means transparent power state management can be implemented

Why use Node.js for IoT?

- Good support for interfacing to native C/C++ libraries
- Community is already using Node.js for embedded devices and robotics
  - For example: http://nodebots.io/, http://cylonjs.com, http://johnny-five.io/…
- A complete end-to-end endpoint/gateway/mobile/browser/server IoT solution is possible
- From Reference [1]

npm

npm is a key packaging utility for most Node.js applications. Here are typical commands:

- `npm install johnny-five`
  - Installs a node.js library, similar to Python's pip
- `npm init`
  - Helps you create a template JSON file (package.json) for documenting packages you create and listing dependencies
- `npm install`
  - The first command you run whenever you clone a Node.js project
- `npm install –save`
  - Adds libraries to an empty project, e.g., npm install --save johnny-five
- `npm config set proxy`
  - Configures npm to use a proxy on an intranet
- `npm run start`
  - Runs scripts; this shortcut must be defined in the package.json

Installing node.js and npm on an RPi3

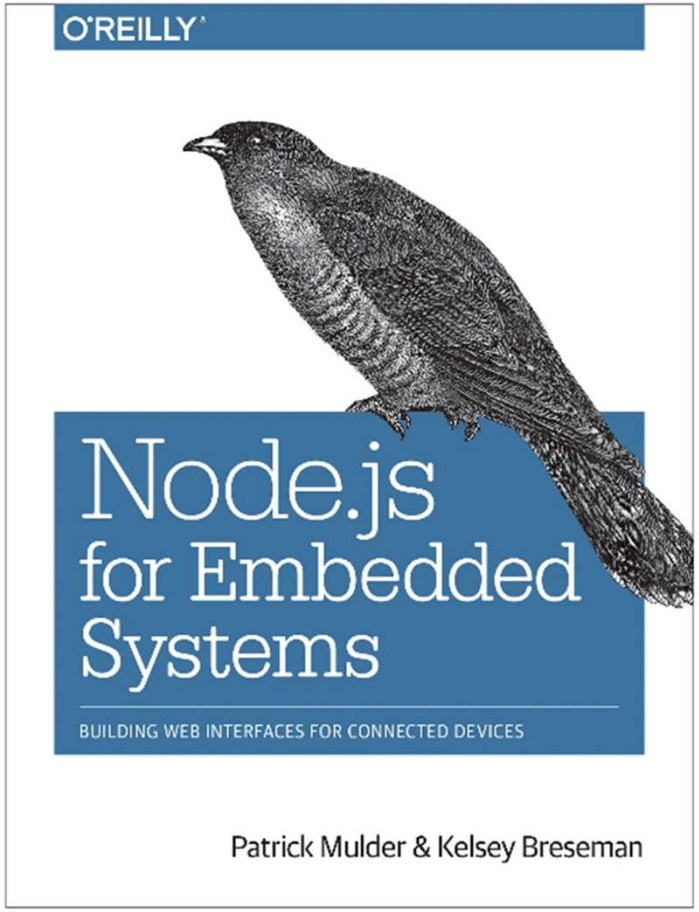
- LTS Node.js version is 10.16.3, npm is 6.11.2
- Latest Node.js version is 12.9.0
- Many different approaches to installing node.js and npm on the Pi – there are other ways besides mine that will work
- Not recommended - using apt-get – apt-get will load from the standard libraries and has a very old version of node (ymmv)
- I recommend using nvm (node version manager) to load both node and npm for either the latest or the stable LTS (Long Term Support) versions

Install nvm, node, and npm

- `curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.34.0/install.sh | bash`
- Restart your terminal
- `nvm –version`
  - Should return a version number like 0.34.0
- `nvm install node`
  - This installs the latest node, 12.9.0
- `nvm install 10.16.3`
  - This installs the stable LTS 10.16.3 node
- Be patient as it builds
- `node -v`
  - If working, returns the version number
- `npm –v`
  - If working, returns the version of npm installed with node.js

Node.js for Embedded Systems

- Mulder & Breseman, 2017, O'Reilly
- IoT, Javascript, Node.js introductions
- Various hardware examples: Arduino, Tessel 2, Photon Particle, RPi, others
- Libraries for sensors, robotics, exploring protocols (HTTP, WebSockets), creating web servers for simple pages and jQuery graphics, MQTT cloud pub/sub, Bluetooth
- Supporting web site for each book chapter – chapter 6 includes details of RPi support [3]



Introduction to JavaScript

- C-like syntax, but ";" are optional.
- Strings (like Python):

```
'this is a string'
"so is this"
```

- Declaring a variable:

```
var my_variable;
```

- Points to note:
  - Types are dynamic
  - All numbers are floats
  - Scope is function, not block
  - Scope is generally static, except for 'this'

Introduction to JavaScript

- Objects and associative arrays are the same:

```
dog['ear'] = 'up';      dog.ear = 'up';
```

- Object prototypes cloned with “new” operator:

```
dog = new Dog();
```

- Functions are values. The ‘this’ variable can be used to refer to the current object:

```
dog.raise_ear = function() {
      this.ear = 'up’;
};
```

- JSON-like syntax for object properties:

```
dog = {
      ear: 'up’;
      raise_ear: function (…) {…};
}
```

Node.js REPL & API

- REPL
  - Read-Eval-Print Loop – essentially the Node.js shell or command line
  - Enter by typing “node”
- API
  - Buffers: Byte Arrays

```
> var buf = new Buffer(4);
<Buffer 50 0a 00 03>
> buf.fill(0);
<Buffer 00 00 00 00>
> buf.writeUint8(0x78, 2);
> console.log(buf)
<Buffer 00 00 78 00>
```

- Streams: For user input (shown) or any general data transfers
- More documentation of Node.js elements at [4]

```
// import stream libraries
var stream = require('stream');
var Stream = stream.Stream;

// create new stream to capture data
var ws = new Stream();
ws.writable = true;

// define write behavior
ws.write = function(data) {
 console.log("input=" + data);
}

// when closing a stream
ws.end = function(data) {
 console.log("bye");
}

// combine stream from input to output
process.stdin.pipe(ws);
```

```
$ node pipe_out.js
hello
input=hello
```

Node.js events and callbacks

- Typical node.js pattern for handling events asynchronously is to use a callback function that gets activated when the event occurs.
- This Adafruit example [5] uses a package called “onoff” to monitor a GPIO-connected button and control a GPIO-connected LED.

```
npm install onoff

// button is attached to pin 17, led to 18
var GPIO = require('onoff').Gpio,
    led = new GPIO(18, 'out'),
    button = new GPIO(17, 'in', 'both');

// define the callback function
function light(err, state) {
  // check the state of the button
  // 1 == pressed, 0 == not pressed
  if(state == 1) {
    // turn LED on
    led.writeSync(1);
  } else {
    // turn LED off
    led.writeSync(0);
  }
}
// pass the callback function
// as the first argument to watch()
button.watch(light);
```

Johnny-Five

- JohnnyFive [6] is a rich Node.js library for direct control of SBCs and robotics
- Johnny-Five creates a object corresponding to the SBC you're talking to:

```
board.on("ready",
function(){});
```

- You can directly toggle board elements from the REPL:

```
>> led13.low();
```

- Here's a 12 line robot control program:

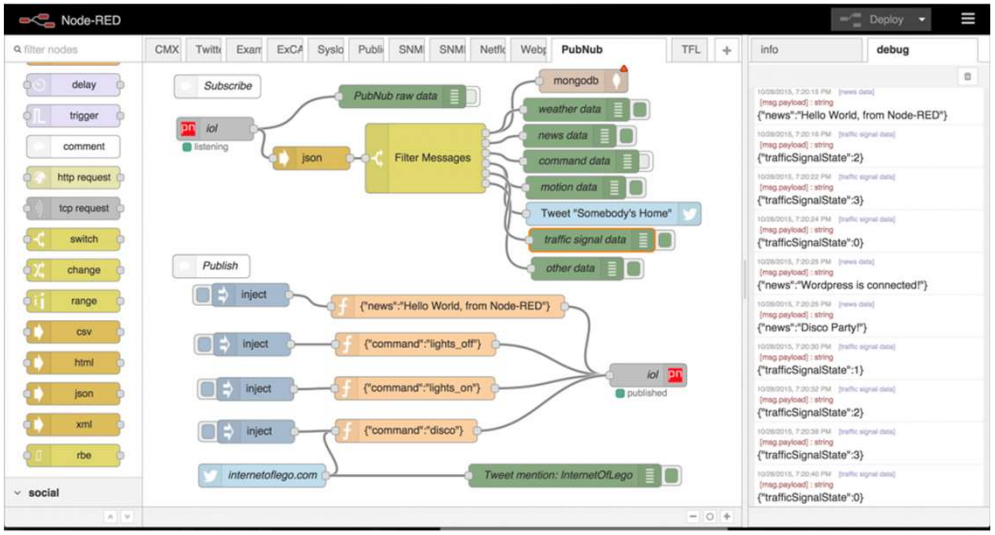
```
// simple_nodebot.js
var five, Nodebot;

five = require("johnny-five");

five.Board().on("ready", function()
{
  Nodebot = new five.Nodebot({
    right: 10,
    left: 11
  });

  this.repl.inject({
    n: Nodebot
  });
})
```

Other Libraries

- Node-serialport – for USB or other serial RX/TX
- Various I2C libraries – sysfs, for example
- LibMRAA – low level bindings to hardware (GPIOs, analog I/O, etc.)
- Temboo – easy integration with cloud services like Twitter
- Cylon.js – jQuery object-oriented style code for hardware interfaces (see code example)
- Many others

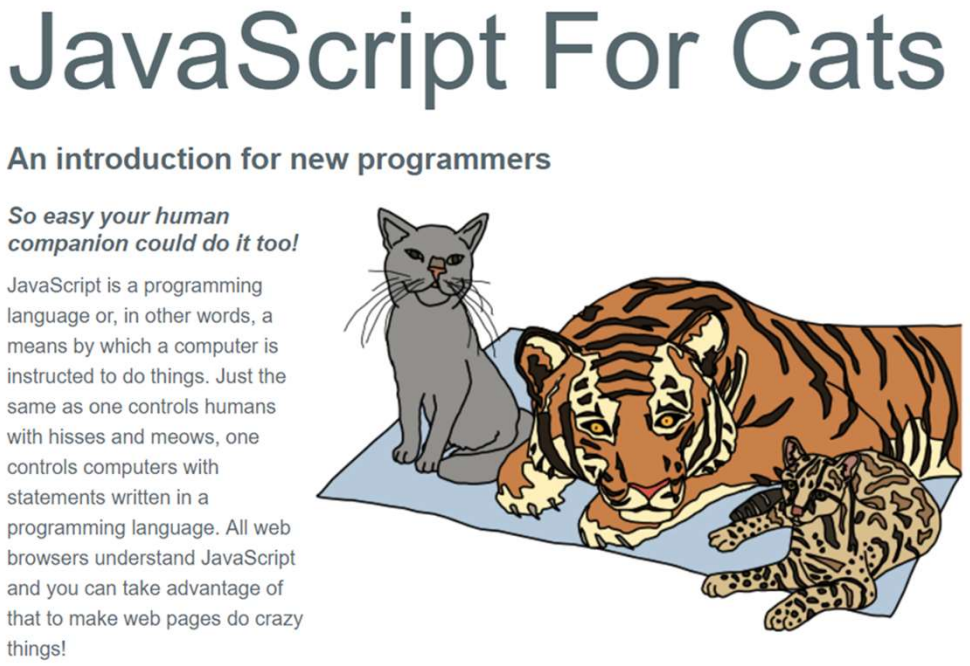
NodeRED

- NodeRED is a visual programming tool built on Node.js [7]
  - Built at IBM and released to GitHub in 2013
  - Works with IBM Bluemix cloud (and others)
- Programs are called flows, steps in the programs are called nodes
- The visual interface for NodeRED is a web GUI served at port 1880
- For more node types [8]



Node & JS References

- Adafruit has a really nice and simple tutorial on Node.js [9]
- Node.js Documentation [10]
- Other Node.js packages (over 130,000 to choose from…) at npm site [11]
- How to get started - list of further reading [12]
- JavaScript for Cats (tutorial) [13]



Node.js thoughts…

- I'm sorry – my opinion – JavaScript is the ugliest language – parentheses, braces, semi-colons, clunky syntax – Python is so much cleaner and clearer
- Now that we're past that – Node.js may be a short path to quick embedded device controls simply because of the rich npm libraries available (like Johnny-Five) – and especially if you have a web/JS background
- If I were building a robot prototype, Node.js is a good first resource
- It's also clearly well supported and has a great open-source community behind it, and it's a clear part of all the cloud vendor toolsets
- NodeRED is interesting, as I have spent time in the past looking at usability of visual programming…
- But I still love my Python…

Next Steps

- If you're on the waitlist, and you want to join the class, keep attending and do all assignments - the waitlists are usually settled in the first two weeks.
  - See Adam Sadoff for questions, I have no control or influence over waitlists!
- Make sure you sign up for Slack and Canvas notifications
- If you're staying in the class, get the Buley and McElroy books
- Distance students staying in the class, contact me about hardware shipments or pickup
- Spend some time reviewing and trying Python and Node.js
- No class on Monday 9/2 – Holiday
- Next class on Wednesday 9/4 – Git, Development, RPis, more…

References

[1] http://cdn.oreillystatic.com/en/assets/1/event/127/Programming%20the%20Internet%20of%20Things%20with%20Node_js%20and%20HTML5%20Presentation.pdf
[2] http://www.npmjs.org
[3] http://embeddednodejs.com/
[4] https://nodejs.org/api/documentation.html
[5] https://learn.adafruit.com/node-embedded-development?view=all
[6] http://johnny-five.io/
[7] https://nodered.org
[8] http://flows.nodered.org
[9] https://learn.adafruit.com/node-embedded-development?view=all
[10] https://nodejs.org/dist/latest-v6.x/docs/api/
[11] https://www.npmjs.org/
[12] http://stackoverflow.com/questions/2353818/how-do-i-get-started-with-node-js
[13] http://jsforcats.com/