Testing and Debugging

Paul O'Fallon
@paulofallon

Outline

- Basic testing with Node’s built-in “assert” module
- More advanced testing with mocha and should.js
- Debugging Node.js apps in Cloud9 IDE

The “assert” module

- Test for (in)equality between expected and actual values
- Test whether a block of code throws (or does not throw) an exception
- Test for the “truthiness” of a value
- Test whether the “error” parameter was passed to a callback
- Each assertion can contain a message to output on failure

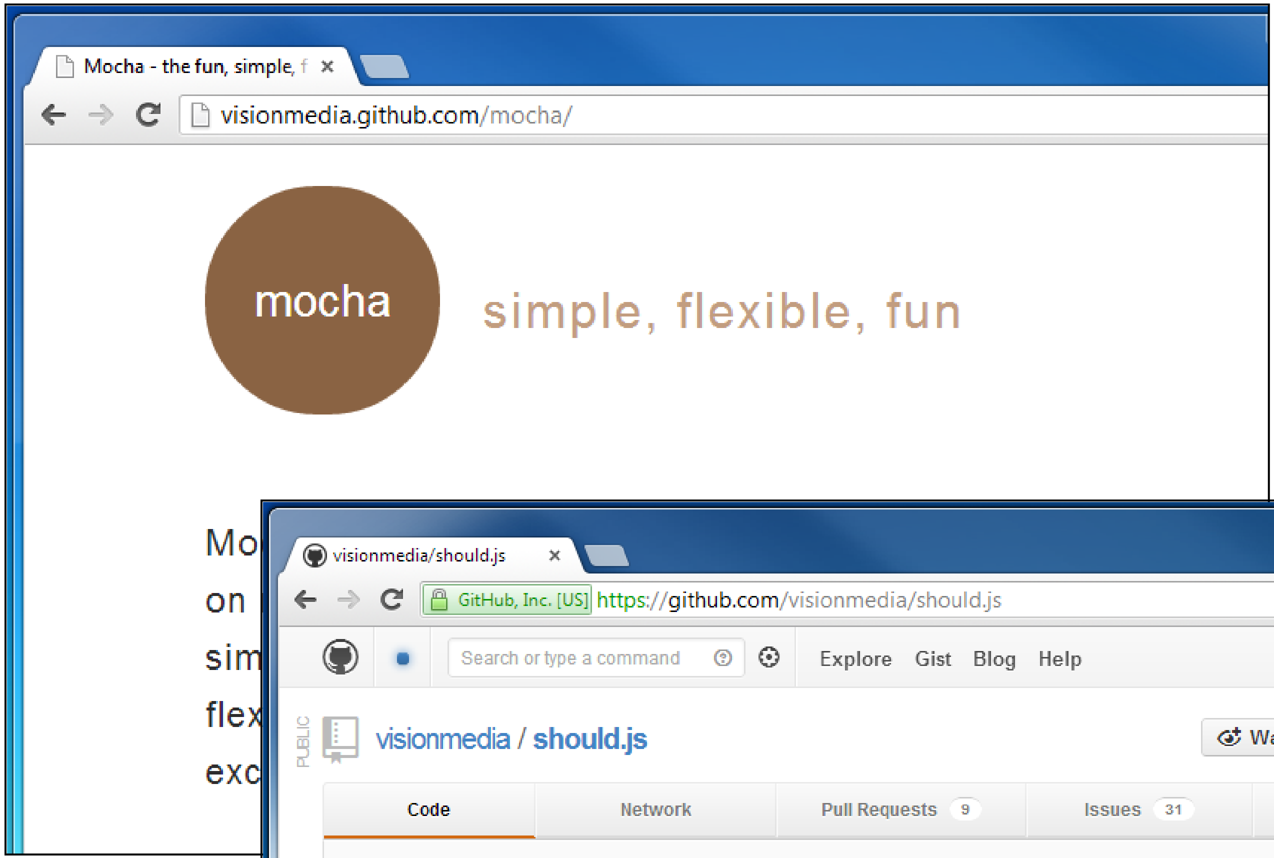
Types of equality

1. **assert.equal(): shallow, coercive equality, as determined by ==**
2. **assert.strictEqual(): strict equality, as determined by ===**
3. **assert.deepEqual():**
   - Identical values are equal (===)
   - Values that are not objects (typeof “object”) are determined by ==
   - Date objects are equal if both refer to the same date/time
   - Other objects (including Arrays) are equal if they have the same number of owned properties, equivalent values for every key and an identical “prototype”

pluralsight

hardcore developer training

Testing with Mocha and should.js

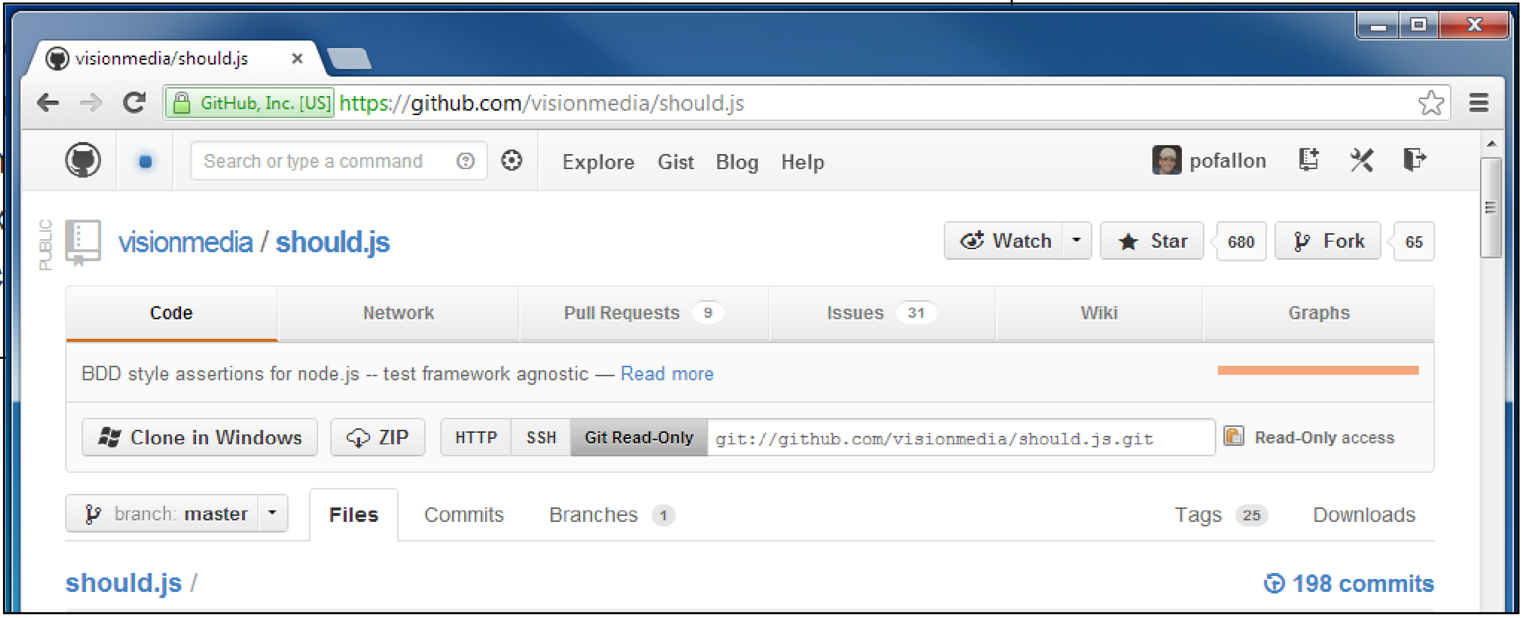


Mocha - the fun, simple, f

visionmedia.github.com/mocha/

simple, flexible, fun

Mo
on
sim
flex
exc



visionmedia/should.js

GitHub, Inc. [US] https://github.com/visionmedia/should.js

visionmedia / should.js

BDD style assertions for node.js -- test framework agnostic — Read more

git://github.com/visionmedia/should.js.git

Read-Only access

should.js /

198 commits

Testing with Mocha

- Runs tests serially (both sync and async tests)
- Test cases are organized into test suites
- Includes before(), after(), beforeEach() and afterEach() hooks
- Support for pending, exclusive and inclusive tests
- Captures test duration, flagging tests that are slow
- Can watch a directory and re-run tests on changes
- Multiple “interfaces” for writing tests (BSD, TDD, Exports, QUnit)
- Multiple “reporters” for rendering test results

Asserting with should.js

Extends Node's "assert" module with BDD style assertions

```
var user = {
  name: 'tj' ,
  pets: ['tobi', 'loki', 'jane', 'bandit']
};
```

extends Object with 'should' function
syntactic sugar for readability
enhanced assertions

```
user.should.have.property('name', 'tj');
```

```
user.should.have.property('pets').with.lengthOf(4);
```

chainable assertions
(inc. volatile properties)

```
someAsyncTask(foo, function(err, result) {

  should.not.exist(err);
  should.exist(result);
```

'should' available statically
(to test for variable existence)

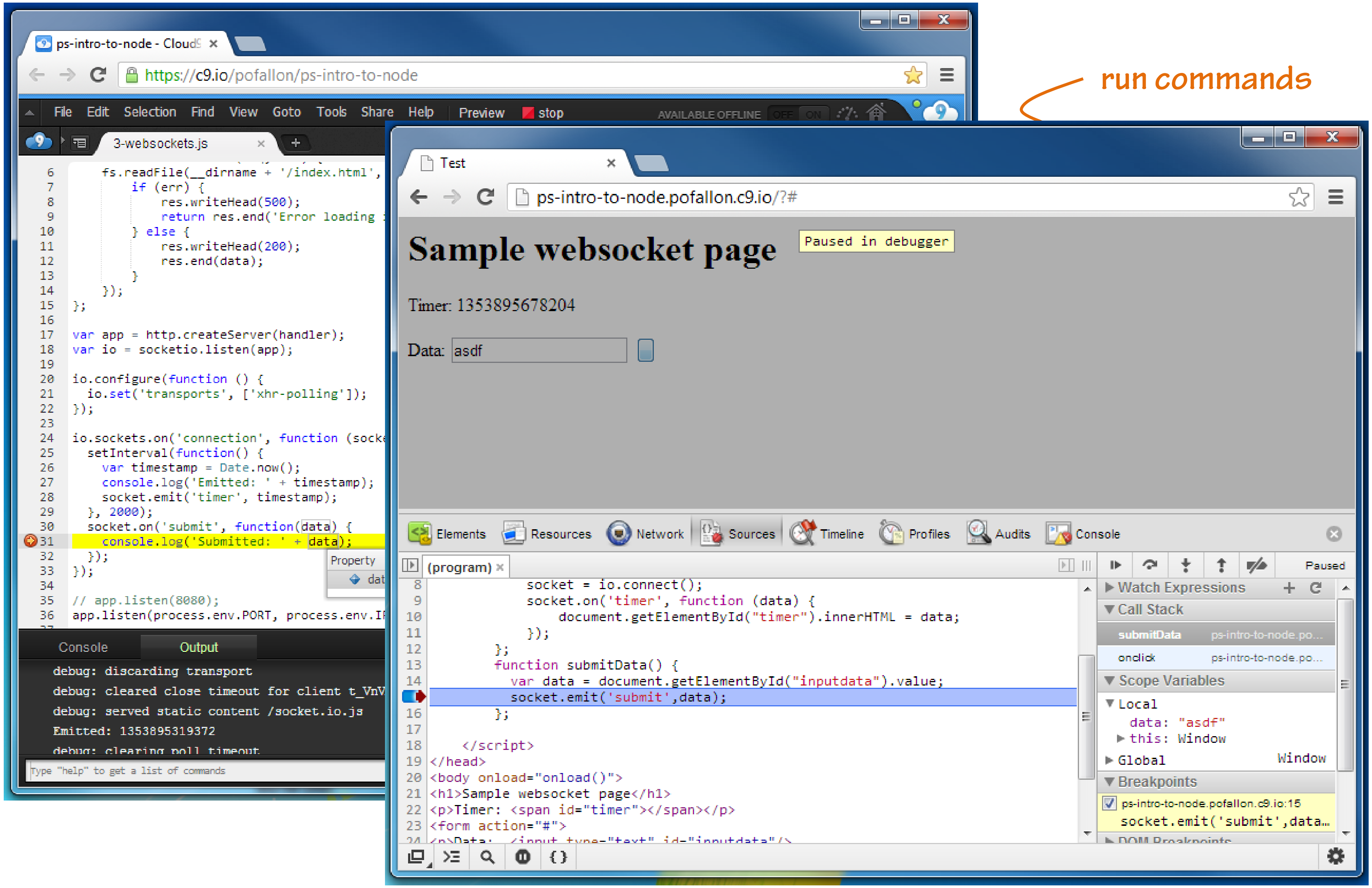
```
  result.bar.should.equal(foo);

});
```

Can assert properties of objects directly

pluralsight
hardcore developer training

Debugging Node with Cloud9

run commands



ps-intro-to-node - Cloud9
https://c9.io/pofallon/ps-intro-to-node
File Edit Selection Find View Goto Tools Share Help Preview stop
AVAILABLE OFFLINE OFF ON
3-websockets.js

```
    fs.readFile(__dirname + '/index.html',
        if (err) {
            res.writeHead(500);
            return res.end('Error loading 
        } else {
            res.writeHead(200);
            res.end(data);
        }
    });
};

var app = http.createServer(handler);
var io = socketio.listen(app);

io.configure(function () {
  io.set('transports', ['xhr-polling']);
});

io.sockets.on('connection', function (socke
  setInterval(function() {
    var timestamp = Date.now();
    console.log('Emitted: ' + timestamp);
    socket.emit('timer', timestamp);
  }, 2000);
  socket.on('submit', function(data) {
    console.log('Submitted: ' + data);
  });
});

// app.listen(8080);
app.listen(process.env.PORT, process.env.IP
```

```
debug: discarding transport
debug: cleared close timeout for client t_VnV
debug: served static content /socket.io.js
Emitted: 1353895319372
debug: clearing poll timeout
```

Type "help" to get a list of commands
Test
ps-intro-to-node.pofallon.c9.io/?#

Sample websocket page

Paused in debugger

Timer: 1353895678204

Data: asdf

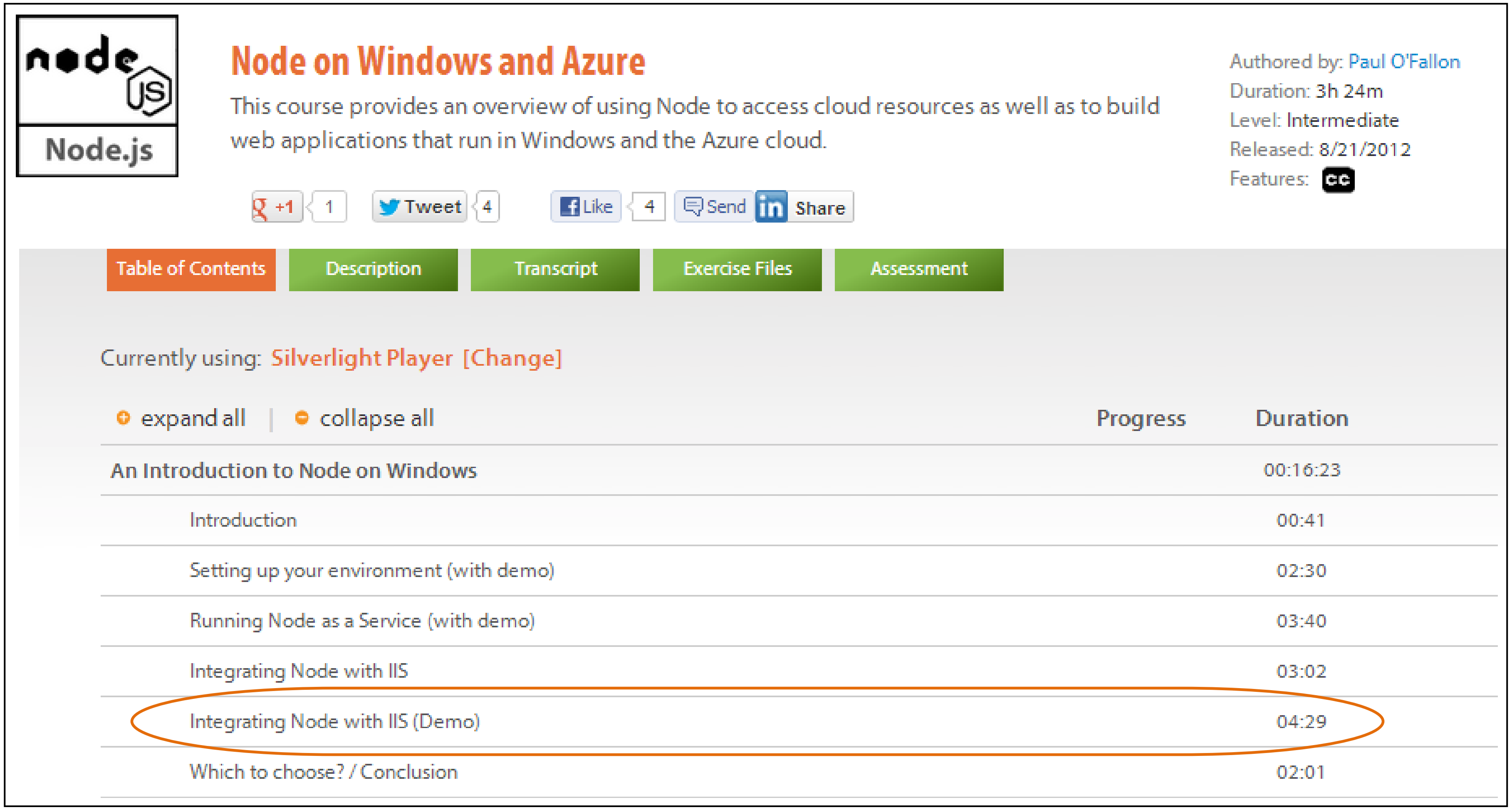
Elements Resources Network Sources Timeline Profiles Audits Console
(program)
Paused

```
            socket = io.connect();
            socket.on('timer', function (data) {
                document.getElementById("timer").innerHTML = data;
            });
        };
        function submitData() {
          var data = document.getElementById("inputdata").value;
          socket.emit('submit',data);
        };

    </script>
</head>
<body onload="onload()">
<h1>Sample websocket page</h1>
<p>Timer: <span id="timer"></span></p>
<form action="#">
<p>Data: <input type="text" id="inputdata"/>
```

Watch Expressions
Call Stack
submitData ps-intro-to-node.po...
onclick ps-intro-to-node.po...
Scope Variables
Local
data: "asdf"
this: Window
Global Window
Breakpoints
ps-intro-to-node.pofallon.c9.io:15
socket.emit('submit',data...
DOM Breakpoints

pluralsight
hardcore developer training

For more on debugging in Node



Node on Windows and Azure

This course provides an overview of using Node to access cloud resources as well as to build web applications that run in Windows and the Azure cloud.

Authored by: Paul O'Fallon
Duration: 3h 24m
Level: Intermediate
Released: 8/21/2012
Features: CC

Table of Contents | Description | Transcript | Exercise Files | Assessment

Currently using: Silverlight Player [Change]

expand all | collapse all

	Progress	Duration
An Introduction to Node on Windows		00:16:23
Introduction		00:41
Setting up your environment (with demo)		02:30
Running Node as a Service (with demo)		03:40
Integrating Node with IIS		03:02
Integrating Node with IIS (Demo)		04:29
Which to choose? / Conclusion		02:01

Conclusion

- **Node’s “assert” module**
- **Testing with Mocha and should.js**
- **Debugging with Cloud9 IDE**

pluralsight

hardcore developer training

References

- CommonJS Unit Testing:
  http://wiki.commonjs.org/wiki/Unit_Testing/1.0
- JS101: Equality (DailyJS): http://dailyjs.com/2012/08/27/equality/
- Node.js Documentation
  http://nodejs.org/api/
- Mocha test framework
  http://visionmedia.github.com/mocha/
- Should.js
  https://github.com/visionmedia/should.js/