

web services with node, express, and hapi

mike cantelon
@mcantelon
mikecantelon.com

eran hammer
@eranhammer
hueniverse.com

Node.js

IN ACTION

Mike Cantelon
TJ Holowaychuk
Nathan Rajlich

 MANNING



outline

- { node

- { express

- { hapi

- { routing

- { middleware

- { authentication

- { validation

```
var http = require('http');
```

```
var http = require('http');
```

```
var server = http.createServer(function (req, res) {
```

```
});
```

```
var http = require('http');

var server = http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'application/json'});
  var result = { version: '1.0.0' };
  res.end(JSON.stringify(result));
});
```

```
var http = require('http');

var server = http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'application/json'});
  var result = { version: '1.0.0' };
  res.end(JSON.stringify(result));
});
```

```
server.listen(3000, localhost);
```



```
var http = require('http');

var server = http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'application/json'});
  var result = { version: '1.0.0' };
  res.end(JSON.stringify(result));
});

server.listen(3000, localhost);
```

```
var http = require('http');
```

```
var handler = function (req, res) {  
  res.writeHead(200, {'Content-Type': 'application/json'});  
  var result = { version: '1.0.0' };  
  res.end(JSON.stringify(result));  
};
```

```
var server = http.createServer(handler);
```

```
server.listen(3000, localhost);
```

```
var http = require('http');

var dispatcher = function (req, res) {
  switch(req.url) {
    case '/version': version(req, res); break;
    case '/user': user(req, res); break;
    default: notFound(req, res); break;
  };
};

var server = http.createServer(dispatcher);

server.listen(3000, localhost);
```

```
var version = function (req, res) {  
  res.writeHead(200, {'Content-Type': 'application/json'});  
  var result = { version: '1.0.0' };  
  res.end(JSON.stringify(result));  
};
```

```
var user = function (req, res) {  
  res.writeHead(200, {'Content-Type': 'application/json'});  
  var result = { id: 23123, name: 'joe' };  
  res.end(JSON.stringify(result));  
};
```

```
var notFound = function (req, res) {  
  res.writeHead(404, {'Content-Type': 'application/json'});  
  var result = { error: 'not found' };  
  res.end(JSON.stringify(result));  
};
```

basic router facilities

- { http methods

- { automatic data marshaling
 - { text, json, html, binary

- { path processing
 - { parameters, query

- { extensibility

framework facilities

- { foundation for collaboration
 - { required in large teams

- { reuse common patterns
 - { middleware, plugins, modules

- { battle tested
 - { sockets, timeouts, memory leaks, stability, debugging

express^{3.0.0}

```
var express = require('express');
```

```
var http = require('http');
```



```
var express = require('express');
```

```
var http = require('http');
```

```
var app = express();
```

```
var express = require('express');
```

```
var http = require('http');
```

```
var app = express();
```

```
app.configure(function () {
```

```
    app.use(app.router);
```

```
});
```

```
var express = require('express');
```

```
var http = require('http');
```

```
var app = express();
```

```
app.configure(function () {
```

```
  app.use(app.router);
```

```
});
```

```
app.get('/version', function (req, res) {
```

```
  res.writeHead(200, {'Content-Type': 'application/json'});
```

```
  var result = { version: '1.0.0' };
```

```
  res.end(JSON.stringify(result));
```

```
});
```

```
var express = require('express');
var http = require('http');

var app = express();

app.configure(function () {
  app.use(app.router);
});

app.get('/version', function (req, res) {
  res.writeHead(200, {'Content-Type': 'application/json'});
  var result = { version: '1.0.0' };
  res.end(JSON.stringify(result));
});

http.createServer(app).listen(3000);
```

```
var express = require('express');
var http = require('http');

var app = express();

app.configure(function () {
  app.use(app.router);
});

app.get('/version', function (req, res) {
  var result = { version: '1.0.0' };
  res.json(result);
});

http.createServer(app).listen(3000);
```

```
var express = require('express');
var http = require('http');

var app = express();

app.configure(function () {
  app.use(app.router);
});

app.get('/version', function (req, res) {
  var result = { version: '1.0.0' };
  res.json(result);
});

http.createServer(app).listen(3000);
```

```
var express = require('express');
var http = require('http');

var app = express();

app.configure(function () {
  app.use(app.router);
});

var version = function (req, res) {
  var result = { version: '1.0.0' };
  res.json(result);
};

app.get('/version', version);

http.createServer(app).listen(3000);
```

```
var user = function (req, res) {  
  
  var item = { id: 1, name: 'steve' };  
  
  res.json(item);  
};  
  
app.get('/user', user);
```



```
var users = require('./users.json');
```

```
var user = function (req, res) {
```

```
    var item = users[req.query.id];
```

```
    res.json(item);
```

```
};
```

```
app.get('/user', user);
```

```
var users = require('./users.json');

var user = function (req, res) {

  var item = users[req.query.id];
  if (!item) return res.send(404, 'not found');
  res.json(item);
};

app.get('/user', user);
```

```
var users = require('./users.json');

var user = function (req, res) {

  var item = users[req.params.id];
  if (!item) return res.send(404, 'not found');
  res.json(item);
};

app.get('/user/:id', user);
```

```
var users = require('./users.json');
```

```
var user = function (req, res) {  
  if (!req.params.id) return res.json(users);  
  var item = users[req.params.id];  
  if (!item) return res.send(404, 'not found');  
  res.json(item);  
};
```

```
app.get('/user/:id?', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
app.configure(function () {
```

```
    app.use(app.router);
```

```
});
```

```
app.get('/user', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
app.configure(function () {  
    app.use(express.bodyParser());  
    app.use(app.router);  
});
```

```
app.get('/user', user);
```

```
var express = require('express');  
var app = express();
```

```
app.configure(function () {  
  app.use(express.bodyParser());  
  app.use(app.router);  
});
```

```
app.get('/user', user);
```

```
var register = function (req, res) {  
  // use req.body to create new user  
};  
app.post('/user', register);
```

```
var express = require('express');  
var app = express();
```

```
app.configure(function () {  
  app.use(express.bodyParser());  
  app.use(app.router);  
});
```

```
app.get('/user', user);
```

```
var register = function (req, res) {  
  // use req.body to create new user  
};  
app.post('/user', register);
```



```
app.configure(function () {  
  app.use(express.bodyParser());  
  app.use(app.router);  
});
```

```
var myMiddleware = function (req, res, next) {  
  res.setHeader('X-API-Version', '1.0.0');  
  return next();  
};
```

```
app.configure(function () {  
  app.use(myMiddleware);  
  app.use(express.bodyParser());  
  app.use(app.router);  
});
```

```
var express = require('express');
```

```
var app = express();
```

```
app.configure(function () {
```

```
    app.use(app.router);
```

```
});
```

```
app.get('/user', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
var validate = function (username, password, callback) {  
    var result = (username === 'steve' && password === '12345');  
    callback(null, result);  
};
```

```
app.configure(function () {
```

```
    app.use(app.router);
```

```
});
```

```
app.get('/user', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
var validate = function (username, password, callback) {  
    var result = (username === 'steve' && password === '12345');  
    callback(null, result);  
};
```

```
var auth = express.basicAuth(validate);
```

```
app.configure(function () {
```

```
    app.use(app.router);
```

```
});
```

```
app.get('/user', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
var validate = function (username, password, callback) {  
    var result = (username === 'steve' && password === '12345');  
    callback(null, result);  
};
```

```
var auth = express.basicAuth(validate);
```

```
app.configure(function () {  
    app.use(auth);  
    app.use(app.router);  
});
```

```
app.get('/user', user);
```

```
var express = require('express');
```

```
var app = express();
```

```
var validate = function (username, password, callback) {  
    var result = (username === 'steve' && password === '12345');  
    callback(null, result);  
};
```

```
var auth = express.basicAuth(validate);
```

```
app.configure(function () {  
  
    app.use(app.router);  
});
```

```
app.get('/user', auth, user);
```

things to try...

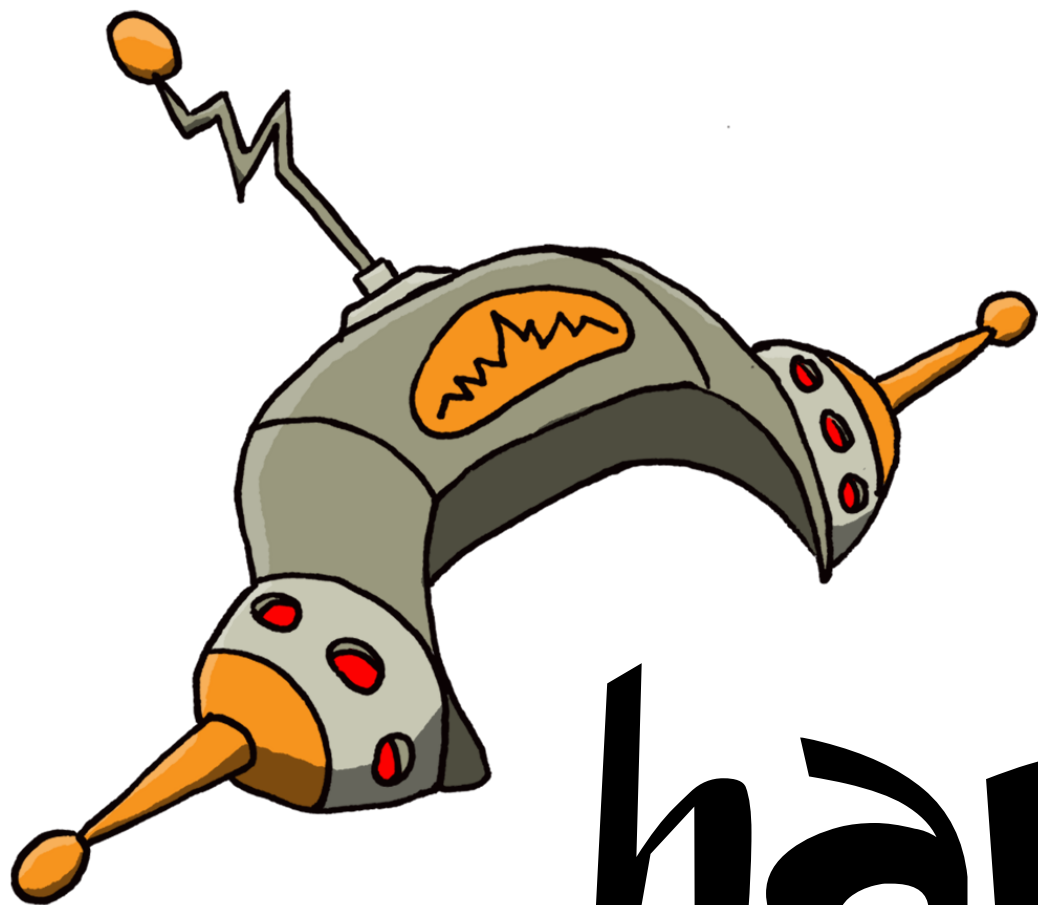
- { start from scratch, see what you remember

- { play with the examples

- { add middleware and play with load order

- { create an endpoint to find users based on criteria

- { use the user.json file to authenticate (static password)



hapi

```
var hapi = require('hapi');
```

```
var hapi = require('hapi');
```

```
var server = new hapi.Server(8000);
```

```
var hapi = require('hapi');
```

```
var server = new hapi.Server(8000);
```

```
var version = function (req) {  
  req.reply({ version: '1.0.0' });  
};
```

```
var hapi = require('hapi');
```

```
var server = new hapi.Server(8000);
```

```
var version = function (req) {  
  req.reply({ version: '1.0.0' });  
};
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var hapi = require('hapi');

var server = new hapi.Server(8000);

var version = function (req) {
  req.reply({ version: '1.0.0' });
};

server.route({
  method: 'GET',
  path: '/version',
  handler: version
});

server.start();
```

```
var user = function (req) {  
  
    var item = { id: 1, name: 'steve' };  
  
    req.reply(item);  
};  
  
server.route({  
    method: 'GET',  
    path: '/user',  
    handler: user  
});
```

```
var users = require('./users.json');
```

```
var user = function (req) {
```

```
    var item = users[req.query.id];
```

```
    req.reply(item);
```

```
};
```

```
server.route({
```

```
    method: 'GET',
```

```
    path: '/user',
```

```
    handler: user
```

```
});
```



```
var users = require('./users.json');
```

```
var user = function (req) {
```

```
    var item = users[req.params.id];
```

```
    req.reply(item);
```

```
};
```

```
server.route({
```

```
    method: 'GET',
```

```
    path: '/user/{id}',
```

```
    handler: user
```

```
});
```

```
var users = require('./users.json');

var user = function (req) {

  var item = users[req.params.id];
  if (!item) return req.reply(hapi.error.notFound());
  req.reply(item);
};

server.route({
  method: 'GET',
  path: '/user/{id}',
  handler: user
});
```

```
var users = require('./users.json');
```

```
var user = function (req) {  
  if (!req.params.id) return req.reply(users);  
  var item = users[req.params.id];  
  if (!item) return req.reply(hapi.error.notFound());  
  req.reply(item);  
};
```

```
server.route({  
  method: 'GET',  
  path: '/user/{id?}',  
  handler: user  
});
```

```
var register = function (req) {  
  // use req.payload to create new user  
};
```

```
server.route({  
  method: 'POST',  
  path: '/user',  
  handler: register  
});
```

```
var register = function (req) {  
  // use req.payload to create new user  
};  
  
server.route({  
  method: 'POST',  
  path: '/user',  
  handler: register,  
  config: {  
    validate: {  
      payload: {  
        name: hapi.types.String().alphanum().max(30).required()  
      }  
    }  
  }  
});
```

```
var server = new hapi.Server(8000);
```

```
var version = function (req) {  
  req.reply({ version: '1.0.0' });  
};
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var server = new hapi.Server(8000);

var version = function (req) {
  req.reply({ version: '1.0.0' })
    .header('X-API-Version', '1.0.0');
};

server.route({
  method: 'GET',
  path: '/version',
  handler: version
});
```

```
var server = new hapi.Server(8000);
```

```
var version = function (req) {  
  req.reply({ version: '1.0.0' });  
};
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
server.ext('onPreResponse', function (request, next) {  
  request.response().header('X-API-Version', '1.0.0');  
  return next();  
});
```



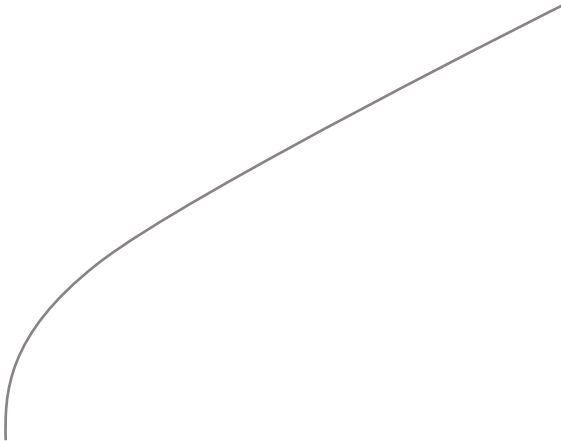
```
var server = new hapi.Server(8000);
```

```
var version = function (req) {  
  req.reply({ version: '1.0.0' });  
};
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
server.ext('onPreResponse', function (request, next) {  
  request.response().header('X-API-Version', '1.0.0');  
  return next();  
});
```

onRequest
onPreAuth
onPostAuth
onPreHandler
onPostHandler
onPreResponse



```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var validate = function (username, password, callback) {  
  var result = (username === 'steve' && password === '12345');  
  callback(null, result, { user: 'steve'});  
};
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var validate = function (username, password, callback) {  
  var result = (username === 'steve' && password === '12345');  
  callback(null, result, { user: 'steve'});  
};
```

```
server.auth('password', {  
  scheme: 'basic',  
  validateFunc: validate  
});
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var validate = function (username, password, callback) {  
  var result = (username === 'steve' && password === '12345');  
  callback(null, result, { user: 'steve'});  
};
```

```
server.auth('password', {  
  scheme: 'basic',  
  validateFunc: validate  
});
```



basic
cookie
hawk
bewit

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var validate = function (username, password, callback) {  
  var result = (username === 'steve' && password === '12345');  
  callback(null, result, { user: 'steve'});  
};
```

```
server.auth('password', {  
  scheme: 'basic',  
  validateFunc: validate,  
  defaultMode: true  
});
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version  
});
```

```
var validate = function (username, password, callback) {  
  var result = (username === 'steve' && password === '12345');  
  callback(null, result, { user: 'steve'});  
};
```

```
server.auth('password', {  
  scheme: 'basic',  
  validateFunc: validate  
});
```

```
server.route({  
  method: 'GET',  
  path: '/version',  
  handler: version,  
  config: { auth: 'password' }  
});
```

things to try...

- { start from scratch, see what you remember

- { play with the examples

- { create an endpoint to find users based on criteria

- { change the validation rules

- { add reverse proxy using the express server as upstream

thank you!