Unit 13.2 Highlights

(Source: http://expressjs.com/)

expressjs - Express is a minimal and flexible Node.js web application framework providing features such as routing
middleware - Software that lies between a client and the backend supporting on it (i.e., databases)

.**listen() -** method which Binds and listens for connections on the specified host and port
.get() - method to handle HTTP GET requests
.post() - method to handle HTTP POST requests
.put() - method to handle HTTP GET requests
.delete() - method to handle HTTP DELETE requests

Ex. 1 (installation, from command line)

```
npm install express --save
```

Ex. 2 (installation, within nodejs application)

```
var express = require("express"); // import express module
var app = express();
var PORT = 3000;

// GET method route
app.get('/', function (req, res) {
  res.send('GET request to the homepage')
})

// POST method route
app.post('/', function (req, res) {
   res.send('POST request to the homepage')
})

app.listen(PORT, function(){
    console.log(`Example app listening on port ${PORT}!`)
});
```

request.params - Property (object) containing properties mapped to the named route “parameters”. If you have the route /user/:name, then the “name” property is available as req.params.name. This object defaults to {}.

Ex. 3

```
var express = require("express"); // import express module
var app = express();
var PORT = 3000;

app.get("/:character", function(req, res) { // import express module
  var chosen = req.params.character;
  res.end(chosen); // prints the value of the parameter entered
});


i.e. HTTP GET localhost:3000/darkseid
```

response.json() - Parses incoming requests and responses, similar to res.send(), but sends responses in JSON format

Ex. 4

```
var character = {
  routeName: "green ranger",
  name: "tommy",
  role: "awesome",
  age: 18,
  forcePoints: 20000000
}

var express = require("express"); // import express module
var app = express();
var PORT = 3000;

// GET method route
app.get('/', function (req, res) {
  res.json(character)
})

app.listen(PORT, function(){
    console.log(`Example app listening on port ${PORT}!`)
});
```

.use() - Method used to configure the middleware used by the routes of the Express HTTP server
body-parser (node module) - Parses incoming request bodies in a middleware before your handlers,
path (node module) - Package provides utilities for working with file and directory paths
path.join() - Path method which joins all given path segments together using the platform specific separator as a delimiter
response.sendFile() - Method used to send HTML files as a response

Ex. 5 (installation, from command line)

```
npm install body-parser --save
```

Ex. 6 (installation, from within node)

```
var express = require("express");
var bodyParser = require("body-parser");
var path = require("path");

var app = express();
var PORT = 3000;

// Sets up the Express app to handle data parsing
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

app.get("/", function(req, res) {
  res.sendFile(path.join(__dirname, "index.html"));
});

// Create New Character with POST - accepts in JSON input
app.post("/api/characters", function(req, res) {
  var newcharacter = req.body; //body-parser module will parse the request body as a JavaScript object
  console.log(newcharacter);
  characters.push(newcharacter);
  res.json(newcharacter);
});
```