

## ExpressJS

Dr. Alfredo Teyseyre  
Dr. Alejandro Zunino  
ISISTAN – UNICEN

1

## Why Express?

- Less code than NodeJS for Web apps:

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

http.createServer(function(request, response) {
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.write("Hello World");
  response.end();
}).listen(3000);
```

```
console.log("Server running on http://localhost:3000");
```

### Where are the paths? (no routing)

3

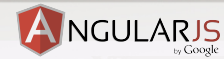
## Routing

- By HTTP method:

- expressApp.**get**(urlPath, requestProcessFunction);
- expressApp.**post**(urlPath, requestProcessFunction);
- expressApp.**put**(urlPath, requestProcessFunction);
- expressApp.**delete**(urlPath, requestProcessFunction);
- expressApp.**all**(urlPath, requestProcessFunction);

5

## The MEAN Stack



Rich frontend framework

**express**

Web dev framework for NodeJS



Event-based concurrency framework



Key-value database

2

## Why Express?

- Less code than NodeJS for Web apps:

```
var express = require("express");
var app = express(); //starts up your app
```

```
app.get("/", function(req, res){
  res.send("Hello world");
});
```

```
app.listen(3000);
console.log("Server started on http://localhost:3000");
```

4

## HttpRequest object

```
expressApp.get('/user/:user_id', function(HttpRequest,
HttpRequest) ...:
```

- request.params - Object containing url route params (e.g. user\_id)
- request.query - Object containing query params
  - http://localhost?foo=9 ⇒ {foo: '9'}
- request.body - Object containing the parsed body
- request.get(field) - Return the value of the specified HTTP header field

6

## httpResponse object

expressApp.get('/user/:user\_id', function(httpRequest, **httpResponse**) ...:

- response.write(content) - Build up the response body with content
- response.status(code) - Set the HTTP status code of the reply
- response.set(prop, value) - Set the response header property to value
- response.end() - End the request by responding to it
- response.end(msg) - End the request by responding with msg
- response.send(content) - Do a write and send

Methods returns the response object so they stack:

- response.status(code).write(content1).write(content2).end();

7

## Middleware: an example

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

```
var myLogger = function (req, res, next) {  
  console.log('Logged:', Date.now());  
  next();  
};
```

```
app.use(myLogger);
```

```
app.get('/', function (req, res) {  
  res.send('Hello World!');  
});
```

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

9

## A simple Web Server

```
var express = require('express');
var app = express(); // Creating an Express "App"
app.use(express.static(__dirname)); // Adding middleware
app.get('/', function (request, response) {
  // A simple request handler
  response.send('Simple web server of files from ' + __dirname);
});
app.listen(3000, function () {
  // Start Express on the requests
  console.log('Listening at http://localhost:3000 exporting the directory ' +
    __dirname);
});
```

11

## Middleware

- Examples:

- Check to see if user is logged in, otherwise send error response and don't call **next()**
- Parse the request body as JSON and attach the object to request.body and call **next()**
- Session and cookie management, compression, encryption, etc.

8

## Middleware

- Give other software the ability to interpose on requests

```
expressApp.all(urlPath, function (request, response, next) {
```

```
// Do whatever processing on request (or setting response)
```

```
next(); // pass control to the next handler
```

```
});
```

- Interposing on all request using the route mechanism

```
expressApp.use(function (request, response, next) {...});
```

10

## Read express Documentation

```
app.use(express.static(__dirname));
```

12

## Fetch from a JSON file

```
var fs = require('fs');
expressApp.get("/object/:objid", function (request, response) {
  var dbFile = "DB" + request.params.objid;
  fs.readFile(dbFile, function (error, contents) {
    if (error) {
      response.status(500).send(error.message);
    } else {
      var obj = JSON.parse(contents); // JSON.parse accepts Buffer types
      obj.date = new Date();
      response.set('Content-Type', 'application/json'); // Same as res.json(obj);
      response.status(200).send(JSON.stringify(obj));
    }
  });
});
Note: Make sure you always call end() or send();
```

13

## DBProduct

```
    "minimum":0,
    "required":true
  },
  "tags":
  {
    "type":"array",
    "items":
    {
      "type":"string"
    }
  }
}
}
```

15

## DBProduct

```
{
  "name":"Product",
  "properties":
  {
    "id":
    {
      "type":"number",
      "description":"Product identifier",
      "required":true
    },
    "name":
    {
      "description":"Name of the product",
      "type":"string",
      "required":true
    },
    "price":
    {
      "type":"number",
```

14

## Exercises

- print debugging information about different requests:
  - ?name=
  - ?name=...?address=...
- connect MongoDB to express and
  - answer an http query by accessing the database previously created

16