

Node.js  
Express.js

# Express.js

“Fast, unopinionated, minimalist web framework for Node.js”

<http://expressjs.com/>

# Εισαγωγή

- Το Express.js αποτελεί είναι node.js framework: έχει γραφτεί 100% πάνω στο node.js
- Το Express περιέχει ένα σύνολο χαρακτηριστικών που διευκολύνουν την ανάπτυξη εφαρμογών όπως: υποστήριξη πολύπλοκης δρομολόγησης, ευκολότερο χειρισμό requests/responses κ.λπ.
- Με απλά λόγια, το Express επιτρέπει την ταχεία ανάπτυξη εφαρμογών node.js: δεν χρειάζεται να ανακαλύψουμε εκ νέου τον τροχό.
- Το Express διευκολύνει την οργάνωση της εφαρμογής μας στην αρχιτεκτονική MVC

- Ας ξεκινήσουμε δημιουργώντας το πρώτο μας project με express
- Αρχικά, μεταβαίνουμε στον φάκελο όπου θα αποθηκεύσουμε το project μας στο terminal, και ξεκινάμε με τη δημιουργία του package.json αρχείου
- Terminal -> npm init
- // συμπληρώνουμε τις λεπτομέρειες για το δυναμικό site μας

# Εγκαθιστούμε το express...

- Δημιουργία ενός app.js αρχείου
- //θα μπορούσαμε να δώσουμε οποιοδήποτε όνομα αλλά συνηθίζεται το app.js
- Θα δημιουργήσουμε μια εφαρμογή που ξεκινά έναν server και «ακούει» σε ένα port για requests
- Check: <http://expressjs.com/en/starter/installing.html>

basic Express app -> starts a server and listens on port 8080 for connection

```
project > JS app.js > app.listen() callback
//import express
const express= require('express');

// create app variable -> assigned result of calling express
// add a wide number of methods to app variable
const app= express();

//start a server
app.listen(8080, () => {
  console.log('Yeah I run');
});
```

# Routing

- Η δρομολόγηση(routing) αναφέρεται στον προσδιορισμό του τρόπου με τον οποίο μια εφαρμογή ανταποκρίνεται σε ένα client request σε ένα συγκεκριμένο endpoint( το οποίο είναι ένα path και μια συγκεκριμένη HTTP request method (GET, POST κ.λπ.)
- Κάθε διαδρομή(route) μπορεί να έχει μία ή περισσότερες handler functions
- Route structure:
- `app.METHOD(PATH, HANDLER)`
  - `app`: instance of express.
  - `METHOD`: HTTP request method, in lowercase.
  - `PATH` : path on the server.
  - `HANDLER` : function executed when the route is matched

# Routing

- Ορίζουμε το uri και το callback function χρησιμοποιώντας τη μέθοδο `get` του `express framework`

```
//Use app.get() to handle GET requests
```

- Τα `arguments` που δέχεται η συγκεκριμένη callback function είναι τα `request/response`
- Το `req` και το `res` είναι τα ίδια ακριβώς που είδαμε στο Nodejs

```
routing
app.get('/', (req, res) => {
})
```

//A route method is derived from one of the HTTP methods, and is attached to an instance of the express class.



# Send Responses

- Θα μπορούσαμε να στείλουμε τώρα όποιο response θέλουμε, για παράδειγμα

```
res.status(200).send('HELLO THERE');
```

//res.status() function-> set HTTP status for the response  
(chainable alias of Node's response.statusCode)

```
res.status(200).json({message: 'HELLO THERE', author: 'Aristea'});
```

Με το express δε χρειάζεται να ορίσουμε ότι το response που στέλνουμε πίσω είναι json. Το καταλαβαίνει από μόνο του....

# HTTP response status codes

- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

# APIs

- Όπως είπαμε και στο παρελθόν το API σημαίνει Διεπαφή προγραμματισμού εφαρμογών
- Το API αποτελεί μία διεπαφή των προγραμματιστικών διαδικασιών που παρέχει ένα λογισμικό-λειτουργικό σύστημα-βιβλιοθήκη ώστε να γίνεται δυνατή η αποστολή αιτημάτων ή/και η ανταλλαγή δεδομένων με άλλα λογισμικά.
- Είναι βασικά ένα κομμάτι λογισμικού που μπορεί να χρησιμοποιηθεί από άλλο λογισμικό ώστε διαφορετικές εφαρμογές να «μιλούν» μεταξύ τους.

# APIs

- web API -> χρησιμοποιείται για την ανάπτυξη διαδικτυακών εφαρμογών
- περιέχει ένα σύνολο από μεθόδους που εκτελούν HTTP requests σε έναν server
- web APIs ανταλλάσσουν δεδομένα σε μορφή JSON ή XML (ή και άλλες)

# APIs

- Θα φτιάξουμε τώρα ένα client-server api
- Αυτό θεωρείται και ο πιο διαδεδομένος τύπος API
- Στην πραγματικότητα, τα API δεν χρησιμοποιούνται μόνο για την αποστολή δεδομένων και φυσικά δε σχετίζονται πάντα με το web development ή τη JavaScript.

# Η αρχιτεκτονική REST

- REST => Representational States Transfer
- Τα REST API, τα οποία είναι apis που ακολουθούν την αρχιτεκτονική REST
- Στην αρχιτεκτονική REST έχουμε clients και servers όπου οι clients στέλνουν αιτήματα στους servers που τα επεξεργάζονται και επιστρέφουν ένα response.

# Η αρχιτεκτονική REST

- Τα REST API, είναι διαφορετικά endpoints(api endpoint ie api/users/) που υποστηρίζουν ένα σύνολο λειτουργιών (μεθόδων) HTTP (Post, get, put, patch, delete)(“endpoint is focused on the URL that is used to make a request”)
- CRUD Ops -> Create Read Update Delete

# Η αρχιτεκτονική REST

- Stateless RESTful API-> αυτό σημαίνει ότι κάθε request θα πρέπει να περιλαμβάνει όλα τις πληροφορίες για να το διαχειριστεί ο server.
- Με άλλα λόγια ο server δε πρέπει να θυμάται προηγούμενα requests!



To be continued...

<https://expressjs.com/>

<https://restfulapi.net/>