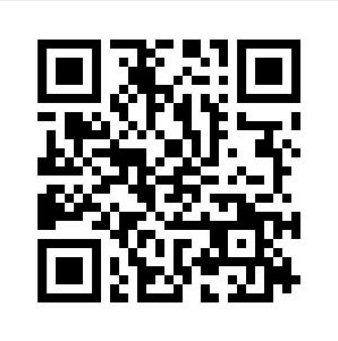
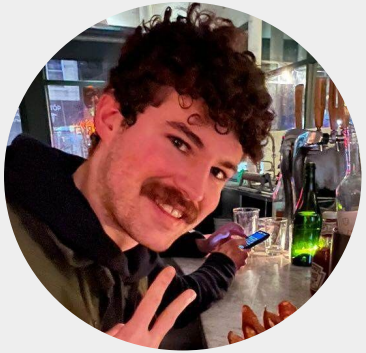


Building Simple APIs

console.log(“I ~~love~~ JavaScript”);

Who am I?



Manuel Dionne

🎓 - University of Waterloo

🏠 - Canada

💼 - Lifion by ADP

What are we doing?

Learning how to make
an API in:

Express

Fast, unopinionated, minimalist web framework for Node.js

‼️⚠️ Prerequisites ⚠️‼️

- Node 18
  - There are obviously other versions.
  - The install varies per OS.
  - Get help installing it here.
- VSCode
  - Also varies per OS
  - Get it here.

In the beginning...

https://github.com/AideTechBot/express-workshop

Initialization 💾

```
$ mkdir project
```

```
$ npm init
```

Optionally:

```
$ git init
```

Sidenote: Linting

```
$ npm i eslint eslint-config-prettier eslint-plugin-prettier
prettier eslint-config-airbnb -D
```

```
$ npx install-peerdeps --dev eslint-config-airbnb
```

Also download the ESLint and Prettier extensions from the VSCode extensions tab!

Sidenote: Linting

⬇️ .eslintrc.json

```
{
    "extends": [
        "airbnb",
        "prettier",
        "plugin:node/recommended"
    ],
    "plugins": [
        "prettier"
    ],
    "rules": {
        "prettier/prettier": "error",
        "no-unused-vars": "warn",
        "no-console": "off",
        "no-process-exit": "off"
    }
}
```

⬇️ .prettierrc.json

```
{
  "printWidth": 120,
  "tabWidth": 2,
  "semi": true,
  "singleQuote": true,
  "bracketSpacing": true,
  "useTabs": false,
  "arrowParens": "always"
}
```

The meat of it 🥩

```
$ npm install express dotenv
```

```
$ npm i -D typescript @types/express @types/node
```

```
$ npx tsc --init
```

Change **compilerOption.outDir** to “./dist”, or whatever you want your output to be.
Also you can set **compilerOption.sourceMap** to true.

The meatier meat of it 🥩🍗🍖

⬇️ index.ts

```
import express from 'express';
import dotenv from 'dotenv';


dotenv.config();
```

The meatier meat of it 🥩🍗🍖

⬇️ index.ts

```
const app = express();

const PORT = process.env.PORT;
```

Baby’s first route

⬇️ index.ts

```
app.get('/', (req , res) => {
  res.send('I responded! Nice.');
});
```

Baby’s first listen

⬇️ index.ts

```
app.listen(PORT, () => {
  console.log(`[server]: Server is running at \
http://localhost:${PORT}`);
});
```

Run it! ...How?

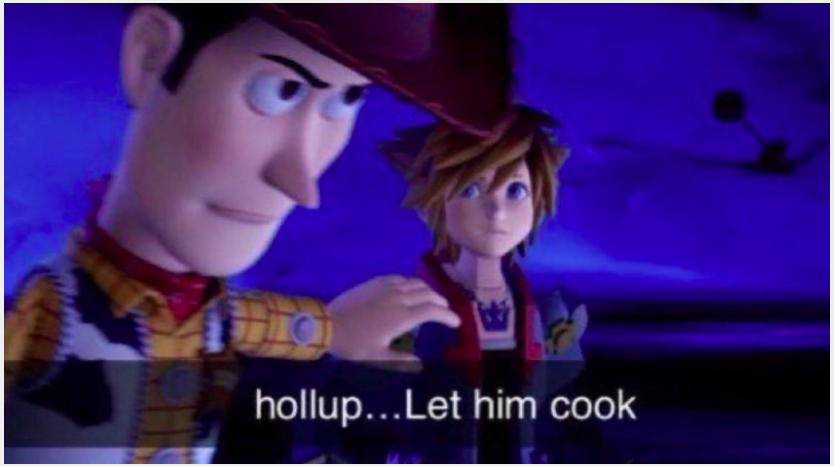
⬇️ package.json

```
"build": "npx tsc",
"start": "node dist/index.js",
```

Test it! ...How?

Download here

What can I cook with this?



Static content

```
// localhost:3000/whatever -> ./public/whatever
app.use(express.static(path.join(__dirname, 'public')));

// localhost:3000/static/whatever -> ./public/whatever
app.use('/static', express.static(path.join(__dirname, 'public')));

// localhost:3000/whatever -> ./public2/whatever
// multiple uses allowed!
app.use(express.static(path.join(__dirname, 'public2')));
```

Sidenote: This is only ok for hackathons (Single threaded, no TLS, HTTP 1, etc.)

Creating routes

```
// GET localhost:3000/
app.get('/', (req , res) => {});
// POST localhost:3000/asdf
app.post('/asdf', (req , res) => {});
// ALL localhost:3000/asdf/ghjkl
app.all('/asdf/ghjkl', (req , res) => {});
// GET localhost:3000/*.exe
app.get(/.*\.exe$/, (req , res) => {});
```

There are other methods like PUT, DELETE, etc.

Query and Route parameters

```
app.get('/:first/:second', (req, res) => {
  const { query, params } = req;
  res.send({ params, query });
});
```

Other useful things about requests

```
const { baseUrl, body, cookies, hostname, ip, path, protocol,
route, secure } = req;
const acceptsHTML = req.accepts('text/html'); // MIME Type
const contentType = req.get('Content-Type'); // Headers
```

Sidenote: body and cookies are undefined by default, you need middleware to parse them.

Response objects

```
res.cookie('Array', { items: [1, 2, 3] }, { maxAge: 900000 });
res.download('/randomfile.exe');
res.status(404);
res.set('Content-Type', 'text/plain');
res.type('html');
res.redirect('http://google.com');
res.json({ hello: 'world' });
res.send('Hello world');
res.sendFile('./asdf.txt');
```

Error handling

```
app.get('/', (req, res) => {
  throw new Error('ouchie owie');
});

app.use((err: any, req: any, res: any, next: any) => {
  console.error(err.stack);
  res.status(500);
  res.send(`Internal Server Error: ${err.message}`);
});
```

Order of execution of **app.whatever()** is important!!

Authentication example

Goal:

Make a simple webpage that you can only access with a session.

Setup

```
npm i express-session
npm i -D @types/express-session
```

```
import sessions from 'express-session';
app.use(
  sessions({
    // DO NOT DO THIS IN PROD
    secret: 'thisisasecretthatshouldnotbestoredinplaintextinprod',
    saveUninitialized: true,
    cookie: { maxAge: ONE_DAY },
    resave: false,
  })
);
```

Serving/parsing

```
// parsing the incoming data
app.use(express.json());
app.use(express.urlencoded({ extended: true }));

//serving the html and css
app.use(express.static('front-end'));
```

Fake database

```
export type UserEntry = {
  password: string; // DO NOT DO THIS
  data: string;
  session: any;
};

const _database: Record<string, UserEntry> = {};
```

Fake database

```
export const getUserEntryById =
    (id: string) => _database?.[id] ?? null;

export const setUserDataById = (id: string, session: any, value: string) => {
  if (_database?.[id]) {
    _database[id].data = value;
    _database[id].session = session;
    return true;
  }
  return false;
};
```

Home endpoint

```
app.get('/', (req, res) => {
  const id = (req?.session as any)?.userId;
  const userEntry = getUserEntryById(id);
  if (userEntry && id) {
    userEntry.session = req?.session;
    res.send(`Welcome ${id}! <a href='/logout'>click to logout</a>`);
  } else {
    res.sendFile('front-end/index.html', { root: __dirname });
  }
});
```

Login endpoint

```
app.post('/login', (req, res) => {
  const { username: id, password } = req.body;
  const userEntry = getUserEntryById(id);
  if (userEntry && password == userEntry.password) {
    userEntry.session = req.session;
    userEntry.session.userId = id;
    console.log(req.session);
    res.send(`Hey there ${id}, welcome <a href=\'/logout'>click to logout</a>`);
  } else {
    res.send('Invalid username or password');
  }
});
```

Misc endpoint

```
app.get('/logout', (req, res) => {
  req.session.destroy(() => {
    res.redirect('/');
  });
});

app.get('/login', (req, res) => {
  res.redirect('/');
});
```

 Thank you! 

Happy hacking!

Resources

Code and slides are on GitHub

Other material referenced:

- Express Examples
- Express API Reference
- How to set up node typescript
- Session management in JS