Node & NPM

📝 2024-03 — ⏱️ 30 min.

Que es NodeJS?

Un *runtime environment* para JS.

Que es un *Runtime Environment*?

Coloquialmente, es donde corre un programa.

Que es NPM?

Node **P**ackage **M**anager

Que es un package?

Un “pequeño” software instalable/agregable a una aplicación de Node que agrega/modifica funcionalidad.

Que es un package?

Todos los packages están listados en https://www.npmjs.com/

Creando un proyecto de Node

```
$ npm init
```

Creando un proyecto de Node - Rapido

```
$ npm init --y
```

Creando un proyecto de Node

```
{
"name": "test",
"version": "1.0.0",
"description": "",
"main": "index.js",
"scripts": {
  "test": "node index"
},
"author": "",
"license": "ISC",
}
```

Express

Un server sencillo

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

function requestListener(req, res) {
  res.writeHead(200);
  res.end("Hello, World!");
}

const server = http.createServer(requestListener);
server.listen(8080);
```

Nodemon

Un package que nos permite hacer "hot reloading", es decir, que al realizar cambios el servidor automaticamente se reinicie.

Nodemon?

Node ya implementó una manera de hacer esto sin instalar packages externos:

```
$ node --watch index.js
```

Sin embargo, Nodemon tiene otras funcionalidades.

Instalar un package

npm i <nombre>

Instalar nodemon

npm i nodemon

npm install nodemon

Responder con JSON

```
const requestListener = function (req, res) {
  res.setHeader("Content-Type", "application/json");
  res.writeHead(200);
  res.end(`{"message": "This is a JSON response"}`);
};
```

JSON

Java **S**cript **O**bject **N**otation

O, Notación de Objeto de JavaScript.

Es una manera de representar (en una string) un objeto de JavaScript.
Tipicamente se espera enviar y recibir datos usando JSON.

Express

npm i express

Inicialización de un servidor de Express

```
const express = require("express");
const app = express();
```

Un Endpoint en Express

```
app.get("/", async function (req, res) {
  res.status(200).json({ message: "Success" });
});
```

Escuchar por peticiones en Express

```
const port = 3000;
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`);
});
```

Operaciones basicas de almacenamiento

Operaciones basicas de almacenamiento

CRUD

Operaciones basicas de almacenamiento

1. **C**
2. **R**
3. **U**
4. **D**

Operaciones basicas de almacenamiento

1. C
2. R
3. U
4. creaDo

Operaciones basicas de almacenamiento

1. C
2. R
3. hUsmear
4. creaDo

Operaciones basicas de almacenamiento

1. aCtualizar
2. R
3. hUsmear
4. creaDo

Operaciones basicas de almacenamiento

1. aCtualizar
2. borRar
3. hUsmear
4. creaDo

Operaciones basicas de almacenamiento

1. Create
2. Read
3. Update
4. Delete

Metodos HTTP

GET	HEAD	POST
PUT	DELETE	CONNECT
OPTIONS	TRACE	PATCH

Metodos HTTP

GET	PUT	POST	PATCH	DELETE

OPTIONS	TRACE	HEAD	CONNECT

Metodos HTTP => CRUD

GET	PUT	POST	PATCH	DELETE
READ	UPDATE/CREATE	CREATE	UPDATE	DELETE

Put vs Patch

Put sobre-escribe (o crea) un recurso.

Patch modifica un recurso ya existente.

Metodos HTTP => CRUD

GET	PUT	POST	PATCH	DELETE
READ	UPDATE/CREATE	CREATE	UPDATE	DELETE

CORS

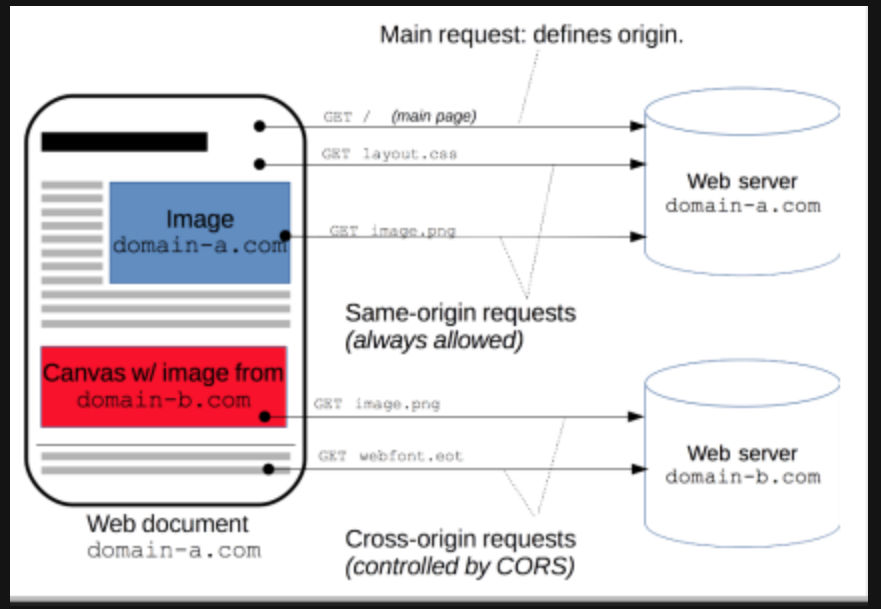
Cross-Origin Resource Sharing

1. Medida de seguridad para prevenir abuso.
2. Previene utilizar recursos de otro origen a menos que este lo permita.

CORS

```
Access to XMLHttpRequest at 'http://localhost:5000/global_config'   step1:1
from origin 'http://localhost:8080' has been blocked by CORS policy:
Response to preflight request doesn't pass access control check: No 'Access-
Control-Allow-Origin' header is present on the requested resource.
```

CORS


Main request: defines origin.
GET / (main page)
GET layout.css
GET image.png
Web server
domain-a.com
Image
domain-a.com
Same-origin requests
(always allowed)
Canvas w/ image from
domain-b.com
GET image.png
GET webfont.eot
Web server
domain-b.com
Web document
domain-a.com
Cross-origin requests
(controlled by CORS)

Metodo OPTIONS + CORS

El metodo OPTIONS tiene como proposito preguntar los permisos de CORS al origen del recurso previo a solicitar el recurso.

CORS

Un problema de CORS 99% de las veces es un problema de backend.

CORS

Un navegador hace un metodo OPTIONS para saber los permisos.

Un cliente REST NO.

Clientes REST como Postman, ThunderClient, Insomnia, etc.

CORS

Front end dev: "Este endpoint me tira error."
Back end dev: "Pero lo pruebo en Postman y me funciona sin problema."

Conclusión: Revisa CORS.

CORS - Solución

```
app.use(function (req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header("Access-Control-Allow-Headers", "*");
  if (req.method === "OPTIONS") {
    res.header("Access-Control-Allow-Methods", "PUT, POST, PATCH, DELETE, GET");
    return res.status(200).json({});
  }
  next();
});
```

Agregar este endpoint al inicio del server responde toda petición de OPTIONS con accesso permitido para todo.

CORS - Solución

```
$ npm i cors
```

Este package hace todo lo que hace el endpoint anterior.

CORS - Solución 2

```
let express = require("express");
let cors = require("cors");
let app = express();

app.use(cors());

app.get("/products/:id", function (req, res, next) {
  res.json({ msg: "This is CORS-enabled for all origins!" });
});
```

"Fallbacks"

```
app.use(async function (req, res) {
  res.status(404).json({ message: "Not found." });
});
```

Este endpoint va al final, para recibir toda petición que no fue servida por otros endpoints.