-Node

JS kan niet als serverside gebruikt worden, maar node kan js laten runnen op server.

Node is not a programming language

Why?

* Non-blocking input/output model
  + Wanneer node op een response wacht kan hij andere dingen doen.
* Lightweight en efficient

Linkerkant blocking non-blocking

Afbeelding met schermafbeelding

Automatisch gegenereerde beschrijving

const fs = require('fs') //fs module

fs.writeFileSync('notes.txt', 'This file was created by Node')

//write some text in textfile

//first name of  file

//second is the data

Tekst will overwritten als er een nieuwe string wordt doorgevoerd.

Toevoegen aan textfile

fs.appendFileSync('notes.txt','\nI\'m 21 years old');

Use other files

APP.JS

const name = require('./utils.js');

console.log(name);

UTILS.JS

const name= 'Cansu';

module.exports = name;

VALIDATOR

npm install validator@10.8.0

console.log(validator.isEmail('test@test.be'))

CHALK

npm install [chalk@2.4.1](mailto:chalk@2.4.1)

const greenMsg = chalk.green.inverse.bold('Succes!');

console.log(greenMsg)

Succres! In groen en dan inverse betekend achtergroend in groen en dan bold

NODEMON

Sudo npm install nodemon@1.18.5 -g

Wordt altijd zelf opnieuw gestart van app.js, en je ziet wat er allemaal gebeurt

INPUT USERS

console.log(process.argv)

[

'/usr/local/bin/node',

'/Users/CansuPakel/Documents/GitHub/1920pop-cansupakel/Node/notesApp/app.js',

'Cansu'

]

YARGS

npm install yargs@12.0.2

To better pase the command line arguments

node app.js add --tile="Things to buy"

console.log(yargs.argv);

{\_: [ 'add' ], tile: 'Things to buy', '$0': 'app.js' }

//custumize yarn version

yargs.version('1.1.0');

YARGS.COMMAND

yargs.command({

  command: "add",

  describe: "Add a new note",

  builder: {

    title: {

      describe: "Note title"

    }

  },

  handler: function(argv) {

    console.log("Adding a new note!", argv);

  }

});

Adding a new note! { \_: [ 'add' ], title: 'Shopping list', '$0': 'app.js' }

title: {

      describe: "Note title",

      demandOption: true //is by default false

    }

notesApp CansuPakel$ node app.js add app.js add

Opties:

--help Toon help [boolean]

--version Toon versie nummer [boolean]

--title Note title [verplicht]

Missend verplichte argument: title

yargs.parse();

Op het einde moet je dit plaatsen anders zal je niets krijgen als output

**JSON**

**const bookJSON = JSON.stringify(book);**

**console.log(bookJSON);**

**const parseData = JSON.parse(bookJSON);**

**console.log(parseData.title)**

**{"title":"Ego us the entity","author":"Ryan"}**

**Ego us the entity**

const fs = require('fs');

// const book = {

//     title: 'Ego us the entity',

//     author: 'Ryan'

// }

// const bookJSON = JSON.stringify(book);

// fs.writeFileSync('1-json.json', bookJSON)

const dataBuffer = fs.readFileSync('1-json.json');

const dataJSON = dataBuffer.toString()

const data = JSON.parse(dataJSON);

console.log(data.title);

**Change value name in 1-json.json**

const dataBuffer = fs.readFileSync("1-json.json");

const dataJSON = dataBuffer.toString();

const user = JSON.parse(dataJSON)

user.name= "test"

const userJSON = JSON.stringify(user);

fs.writeFileSync('1-json.json',userJSON)

**ASYNCHRONOUS basics**

Noun blocking js: continue while is waiting to complete an IO operation.

console.log('Starting')

setTimeout(()=>{

    console.log('2 second timer')

},2000)

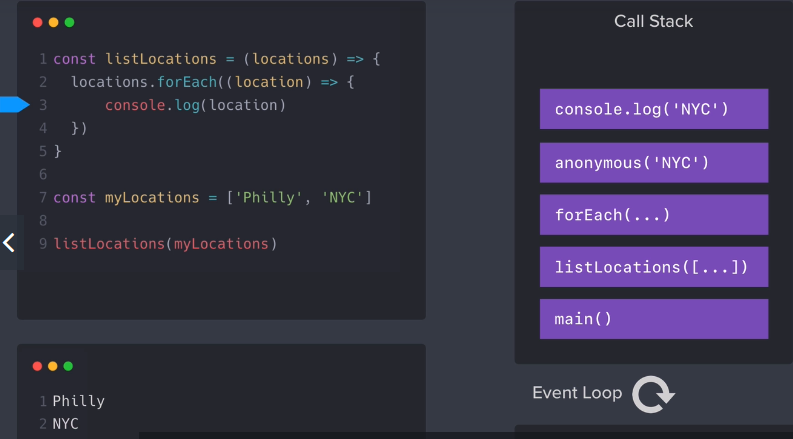
console.log('Stopping')

=>node app.js

Starting

Stopping

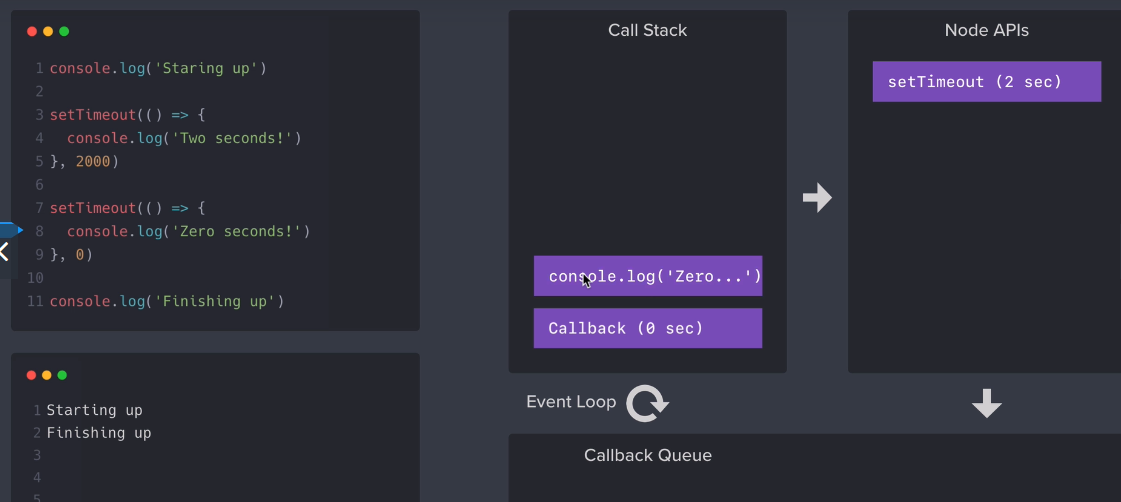
2 second timer



Single threaded, we wachten niet op de node api, we doen gewoon verder.

Callback queue when the call back functions is ready for execution.

Eventloop kijkt naar call stavck en event loop. Kijkt wanneer call stack leeg is. Dus ook geen main()



API

Npm init

npm install [request@2.88.0](mailto:request@2.88.0)

darksky

Geocoding mapbox

const url = 'https://api.darksky.net/forecast/636329bc988959038d08e478e8100926/51,5?units=si&lang=nl'

//units=si in celcius

request({url:url, json:true}, (error, response)=>{

    if(error){

        console.log('Error 443')

    }else if(response.body.error){

        console.log('Unable to find location 400')

     }else {

        console.log(response.body.daily.data[0].summary+ ' Het is '+response.body.currently.temperature + 'graden. Er is ' +  response.body.currently.precipProbability + ' % kans op regen'  )

    }

})

//GEOcoding

const geocodeURL = 'https://api.mapbox.com/geocoding/v5/mapbox.places/Belgium.json?access\_token=pk.eyJ1IjoiY2Fuc3VwYWtlbCIsImEiOiJjanBjeDl4c2owZnRpM2twbGxoNDRxbTZ1In0.qkaxfimkknitlDCrxz\_M9Q';

request({url:geocodeURL, json:true}, (error, response)=>{

    if(error){

        console.log('Unable to connect')

    }else if(response.body.features.length === 0){

        console.log('Unable to find location. Try another one')

    }else{

        const latitude = response.body.features[0].center[1]

        const longitude = response.body.features[0].center[0]

        console.log(latitude, longitude)

    }

})

CALLBACK function

const add = (a,b,callback) => {

    setTimeout(()=> {

        callback(a+b)

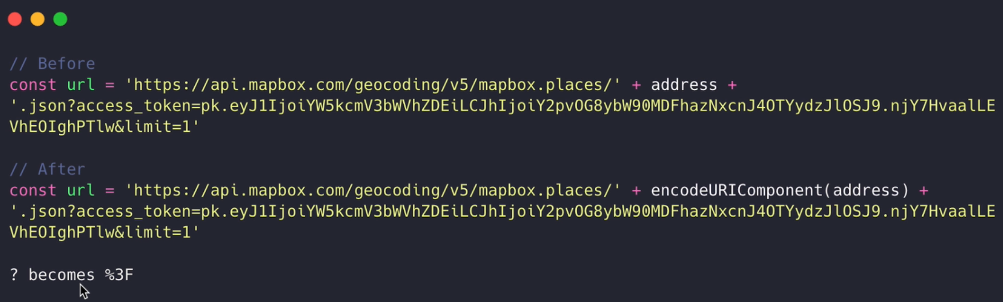
    },2000)

}

 add(1,4, (sum) => {

     console.log(sum)

 })



OBJECT DESTRUCTING

//object destructering

const product = {

    label:'Notebook',

    price: 5,

    stock: 20,

    salePrice: undefined,

    rating:4.2

}

const {label: productLabel, stock, rating=5} = product

console.log(productLabel)

console.log(stock)

console.log(rating)

const transaction = (type,{ label,stock, price }) => {

   console.log(type,label,stock, price)

}

transaction('order',product)

EXPRESS: make it easy to create an webserver.

npm i [express@4.16.4](mailto:express@4.16.4)

nodemon src/app.js => als je een wijziging brengt toont het meteen. Anders moet je steeds xwebserver opnieuw starten.

The file C:\Users\Cansu\AppDat

a\Roaming\npm\nodemon.ps1 is not digitally signed. You

cannot run this script on the current system…..

PS => Set-ExecutionPolicy remotesigned

Template engine Handle bar:

dynamic documents & create code that we can reuse.

npm install [hbs@4.0.1](mailto:hbs@4.0.1)

**nodemon src/app.js -e js,hbs for to run partials**

**QUERY:**

    console.log(req.query) // pakt alles na ? teken

**REQUEST**

**npm i** [**request@2.88.0**](mailto:request@2.88.0)

**HEROKU**

**SSH keys!!**

**Heroku keys:add**

**heroku create weather-application-pop-node**

  "scripts": {

    "start": "node src/app.js"

  },

const port = process.env.PORT || 3000

**git init**

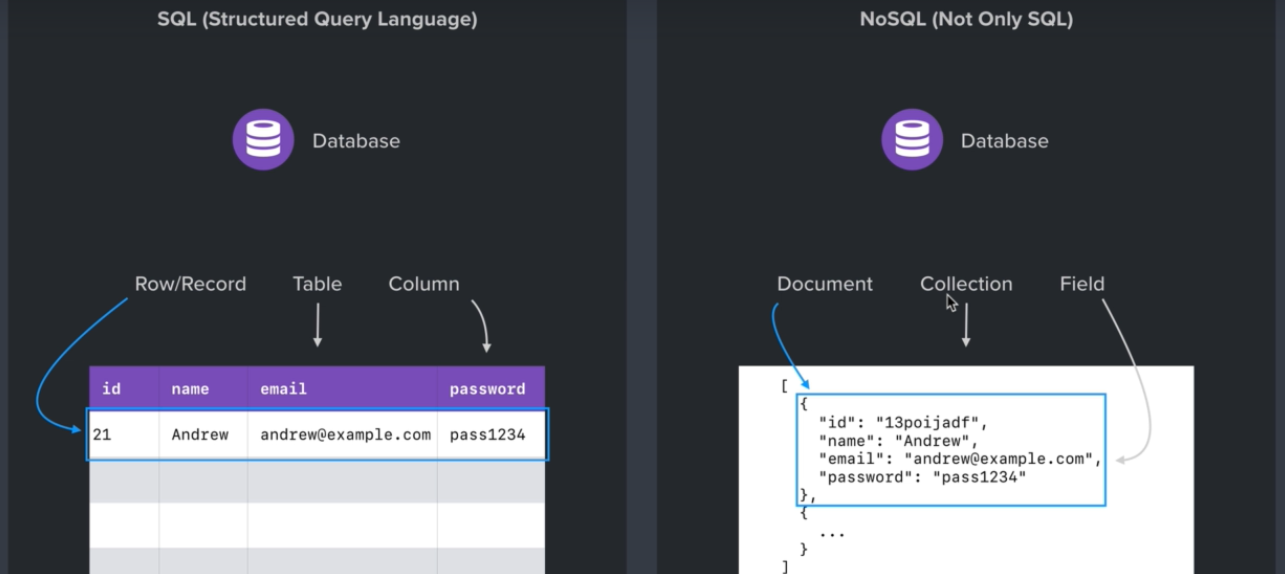
**heroku git:remote -a weather-application-pop-node**

**git add .**

**git commit -m “setup”**

**git push heroku master**

**MONGODB**



**/Users/Cansu/mongodb/bin/mongod.exe --dbpath=/Users/Cansu/mongodb-data**

**Robo 3T => GUI Mongodb**

**npm i** [**mongodb@3.1.10**](mailto:mongodb@3.1.10)

    db.collection('users').insertOne({

        name:'Cansu',

        age:21

    }, (error,result) =>{

        if(error){

            return console.log('Unable to insert user')

        }

        console.log(result.ops) //array of documents

    }) //when it is complete

   db.collection('tasks').insertMany([

        {

            description: 'Nodejs course',

            completed: false

        },{

            description:'React course',

            completed:true

        }

    ], (error, result) => {

        if(error){

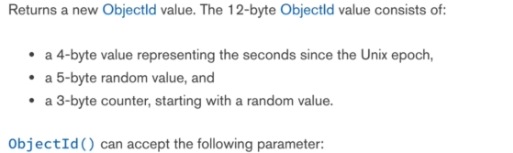
            return console.log(error)

        }

        console.log(result.ops)

    })

**OBJECTID**



**FIND**

db.collection('tasks').findOne({\_id: new ObjectID('5deb863fe08566693c2d24cb')}, (error,result) => {

        if(error){

            return console.log(error)

        }

        console.log(result)

    })

    db.collection('tasks').find({completed:false}).toArray((error,result)=>{

        console.log(result)

    })

**PROMISES**

const doWorkPromise = new Promise((resolve,reject)=>{

    resolve([7,4,1])

 //   reject('Wrong')

})

//only execute wanneer het goed verloopt

doWorkPromise.then((result)=> {

    console.log('Succes',result)

}).catch((error)=>{

    console.log('Error',error)

})

**UPDATE DOCUMENT**

   const updatePromise= db.collection('users').updateOne({

        \_id: new ObjectID('5deb846cf07f642004ffcb12')

    },{

        $set: {

            name: 'Mike'

        }

    }).then((result)=>{

        console.log(result)

    }).catch((error)=>{

        console.log(error)

    })

    const updatePromise= db.collection('users').updateOne({

        \_id: new ObjectID('5deb846cf07f642004ffcb12')

    },{

        $inc: {

            age:1

        }

    }).then((result)=>{

        console.log(result)

    }).catch((error)=>{

        console.log(error)

    })

**DELETE**

    db.collection('users').deleteMany({

        age:23

    })

    db.collection('users').deleteOne({

        name:'Test'

    })

**MONGOOSE**

**npm i** [**mongoose@5.3.16**](mailto:mongoose@5.3.16)

**VALIDATOR**

**npm i** [**validator@10.9.0**](mailto:validator@10.9.0)

**NODEMON EN EXPRESS**

**npm i nodemon@1.18.9 --save-dev**

**npm i** [**express@4.16.4**](mailto:express@4.16.4)

**BCRYPTJS**

**npm i** [**bcryptjs@2.4.3**](mailto:bcryptjs@2.4.3)

**JSONWEBTOKENS**

**npm i** [**jsonwebtoken@8.4.0**](mailto:jsonwebtoken@8.4.0)

**FILEUPLOADS:**

**Npm I** [**multer@1.4.1**](mailto:multer@1.4.1)

**Save it as bytes in user.**

**<img src="data:image/jpg;base64, bytes >**

[**https://jsbin.com/?html,output**](https://jsbin.com/?html,output)

**SHARP: RESIZE IMAGE:**

**Npm install** [**sharp@0.21.1**](mailto:sharp@0.21.1)

**SENDGRID:**

**Create account!**

npm install @sendgrid/mail@6.3.1

//sgMail.send({

//     to:'cansupakel6@gmail.com',

//     from:'cansupakel6@gmail.com',

//     subject:'test',

//     text:'Actie'

// })

**ONLY LOCAL get environment variables:**

**Npm install** [**env-cmd@8.0.2**](mailto:env-cmd@8.0.2) **–save-dev**

**PRODUCTION**

**Create mongodb atlas account**

**taskApp**

**eUnReF2Kj6n2Pdd**

**mongodb+srv://taskApp:<password>@cluster0-7w5fj.mongodb.net/test**

**HEROKU**

**Git init**

**Git status**

**Git add .**

**Git commit -m “dd”**

**heroku create task-app-node-pop**

**Heroku config:set key=value**

**heroku config:set JWT\_SECRET=FGJDFOGJFDOGKSF SENDGRID\_API\_KEY=SG.ePnJpAevSeCCqZp\_NRDMiw.jISZ2cA3Iv2QMJVoSJCaCdm2iznGEWgOLsGqa9bCSjo**

**heroku config:set DB\_URL="mongodb+srv://taskApp:eUnReF2Kj6n2Pdd@cluster0-7w5fj.mongodb.net/task-manager-api?retryWrites=true"**

**git push heroku master**

**TESTEN**

**npm I** [**jest@23.6.0**](mailto:jest@23.6.0) **–save-dev**

**Waarom test?**

* **Het bespaart u tijd**
* **Geeft flexibiliteit voor developers** 
  + **Refactoring**
  + **Profiling (dat sneller gaat)**
  + **Collaboration (als iemand een feature erinsteekt dan na een test zal hij zeker zijn dat hij het niet verpest heeft)**

**SUPERTEST**

**npm install supertest@3.4.1 --save-dev**

**CHAT**

**Npm init**

**npm i** [**express@4.16.4**](mailto:express@4.16.4)

**src => index.js**

const express = require("express");

const path = require("path");

const app = express();

const port = process.env.PORT || 3000;

const publicDirectory = path.join(\_\_dirname, "../public");

app.use(express.static(publicDirectory));

app.listen(port, () => {

  console.log("Server is up", port);

});

**npm i nodemon@1.18.7 --save-dev**

**Package.json**

 "start": "node src/index.js",

    "dev": "nodemon src/index.js"

WEBSOCKET protocol

Ga ik gebruiken, om realtime applicatie te maken met node.js

* Fulll-duplex communication
* Persistent connectie tussen client en server

npm i [socket.io@2.2.0](mailto:socket.io@2.2.0)

INDEX.JS

const express = require("express");

const http = require("http");

const path = require("path");

const app = express();

const socketio = require("socket.io");

const server = http.createServer(app);

const io = socketio(server);

const port = process.env.PORT || 3000;

const publicDirectory = path.join(\_\_dirname, "../public");

app.use(express.static(publicDirectory));

let count = 0;

io.on("connection", socket => {

  console.log("new websocket connection");

  socket.emit("countUpdated", count);

});

server.listen(port, () => {

  console.log("Server is up", port);

});

CHAT.JS aan clientcide (public/js)

const socket = io();

socket.on("countUpdated", count => {

  console.log("count updated", count);

});

CLIENT NAAR SERVER

Index.js

io.on("connection", socket => {

  console.log("new websocket connection");

  socket.emit("countUpdated", count);

  socket.on("increment", () => {

    count++;

    socket.emit("countUpdated", count);

  });

Chat.js

document.querySelector("#increment").addEventListener("click", () => {

  socket.emit("increment");

});

EVERY SINGLE CONNECTION

io.on("connection", socket => {

  console.log("new websocket connection");

  socket.emit("countUpdated", count);

  socket.on("increment", () => {

    count++;

    // socket.emit("countUpdated", count);

    io.emit("countUpdated", count);

  });

});

npm i [bad-words@3.0.0](mailto:bad-words@3.0.0)

CALLBACK FUNCTION only you en niet naar andere users

Index.js van server

socket.on("sendLocation", (coords, callback) => {

    io.emit(

      "message",

      `https://google.com/maps?q=${coords.latitude},${coords.longitude}`

    );

    callback();

  });

Chat.js van client

document.querySelector("#send-location").addEventListener("click", e => {

  e.preventDefault();

  if (!navigator.geolocation) {

    return alert("Geolocation is not supported by your browser.");

  }

  navigator.geolocation.getCurrentPosition(position => {

    socket.emit(

      "sendLocation",

      {

        latitude: position.coords.latitude,

        longitude: position.coords.longitude

      },

      () => {

        console.log("Location shared");

      }

    );

  });

});

Zal het disablen als het message is verzonden. Zodat je niet twee keer verstuurd perongelijk

Daarom worden er callbacks gebruikt.

Mustache library is voor dynamische html

Qs: zal querystring parsen voor rooms en names

Socket.emit => event naar spcefieke client

Io.emit => alle clients

Socket.broadcast.emit => alle client behalve zichzelf

Io.to.emit => everybody in specifiek room

Socket.broadcast.to.emit => everybody in specifiek room behalve zichzelf