



NodeJS Dasar

Pertemuan 2



Sebelum Belajar

- Sudah menyelesaikan Kelas Roadmap JavaScript dari Programmer Zaman Now



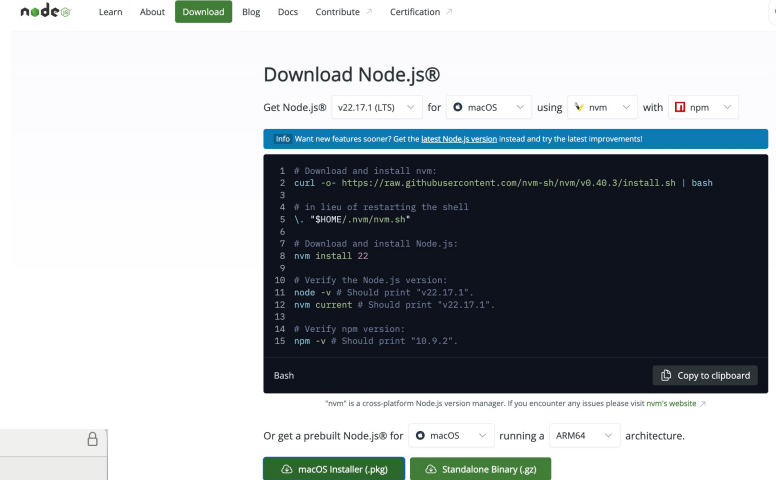
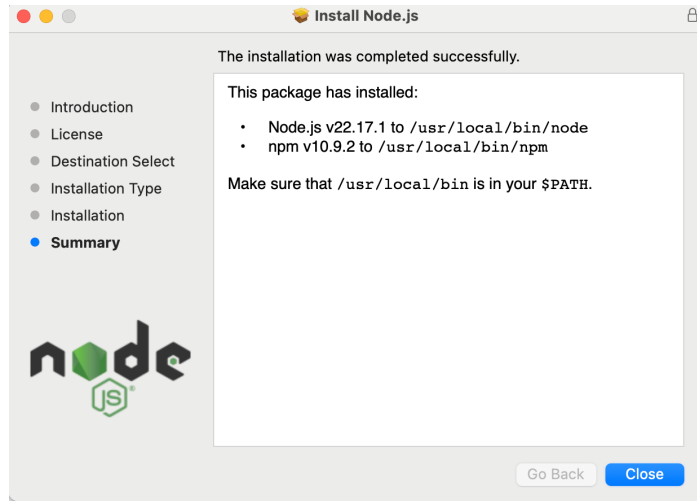
Agenda

- Pengenalan NodeJS
- Pengenalan Concurrency
- NodeJS Architecture
- Menginstall NodeJS
- NodeJS REPL
- Standard Library
- Dan lain-lain

Menginstall NodeJS

Menginstall NodeJS Manual

- Download versi NodeJS LTS (Long Term Support)
- <https://nodejs.org/en/download/>





Menginstall NodeJS dengan Package Manager

- <https://nodejs.org/en/download>
- <https://github.com/nvm-sh/nvm>
- <https://community.chocolatey.org/packages/nodejs>
- <https://formulae.brew.sh/formula/node>



Setting PATH NodeJS

- Setelah menginstall NodeJS, disarankan melakukan setting PATH NodeJS pada sistem operasi kita
- Hal ini agar mudah ketika kita mengakses program NodeJS menggunakan terminal / command prompt

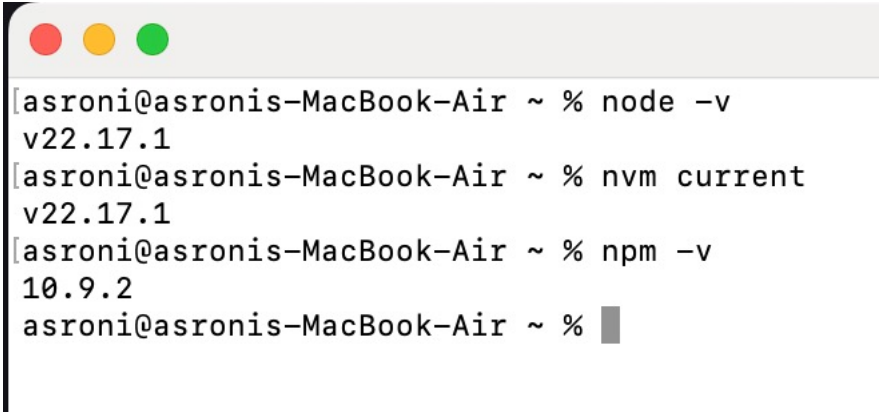


Kode : Mengecek NodeJS

```
→ ~ node --version
```

```
v16.13.0
```

```
→ ~ █
```



```
[asroni@asronis-MacBook-Air ~ % node -v  
v22.17.1
```

```
[asroni@asronis-MacBook-Air ~ % nvm current  
v22.17.1
```

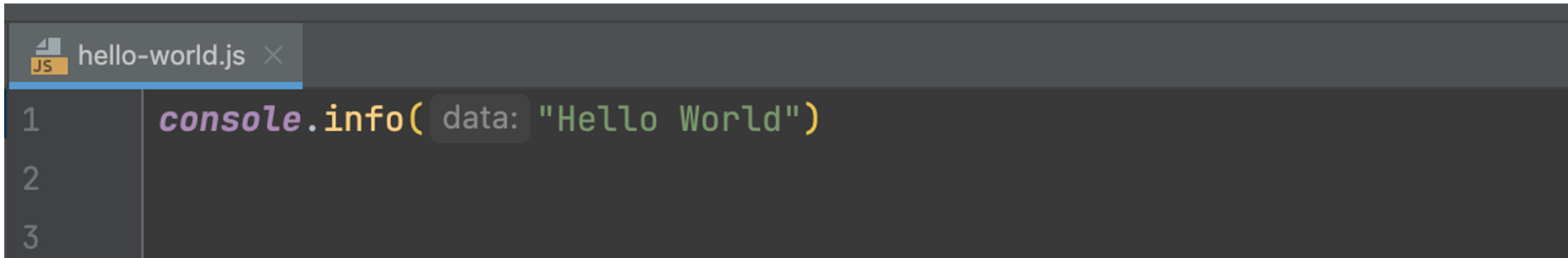
```
[asroni@asronis-MacBook-Air ~ % npm -v  
10.9.2
```

```
asroni@asronis-MacBook-Air ~ % █
```

Hello World



Kode : Hello World



```
JS hello-world.js ×  
1 console.info(data: "Hello World")  
2  
3
```



Menjalankan Kode JavaScript

- Karena NodeJS tidak memerlukan Web Browser, jadi kita bisa langsung menjalankan program JavaScript kita menggunakan aplikasi NodeJS lewat terminal / command prompt, dengan perintah :
- `node namafile.js`



Kode : Menjalankan Hello World

```
→ belajar-nodejs-dasar node hello-world.js
```

```
Hello World
```

```
→ belajar-nodejs-dasar
```

```
● asroni@asronis-MacBook-Air belajar-nodejs-dasar-main % node hello-world.js
```

```
Hello World
```

```
○ asroni@asronis-MacBook-Air belajar-nodejs-dasar-main % █
```

NodeJS REPL



REPL (Read Eval Print Loop)

- REPL singkatan dari Read Eval Print Loop
- Yaitu mekanisme dimana program bisa membaca langsung kode program yang diketikkan, lalu mengeksekusinya, menampilkan hasilnya, lalu mengulangi dari awal lagi
- NodeJS mendukung REPL, sehingga lebih mudah ketika belajar
- Namun tetap, saya menyarankan menyimpan kode program di file JavaScript, agar lebih mudah diubah ketika terjadi masalah
- Untuk menggunakan NodeJS REPL, cukup jalankan aplikasi node saja



Kode : REPL

```
→ ~ node
```

```
Welcome to Node.js v16.13.0.
```

```
Type ".help" for more information.
```

```
> .help
```

```
.break    Sometimes you get stuck, this gets you out
```

```
.clear    Alias for .break
```

```
.editor   Enter editor mode
```

```
.exit     Exit the REPL
```

```
.help     Print this help message
```

```
.load     Load JS from a file into the REPL session
```

```
.save     Save all evaluated commands in this REPL session to a file
```

```
Press Ctrl+C to abort current expression, Ctrl+D to exit the REPL
```

```
> █
```

NodeJS Standard Library



NodeJS Standard Library

- Saat kita belajar JavaScript, di Web Browser, terdapat fitur-fitur yang bernama Web API
- <https://developer.mozilla.org/en-US/docs/Web/API>
- Kebanyakan fitur Web API hanya berjalan di Web Browser, sehingga tidak bisa jalan di NodeJS
- NodeJS sendiri hanya menggunakan bahasa pemrograman JavaScript nya, namun tidak mengadopsi fitur Web API nya, karena itu hanya berjalan di Web Browser
- NodeJS sendiri memiliki standard library yang bisa kita gunakan untuk mempermudah pembuatan aplikasi
- <https://nodejs.org/dist/latest-v16.x/docs/api/>

```
❌ asroni@asronis-MacBook-Air belajar-nodejs-dasar-main % node web-api.js
/Users/asroni/Documents/Kuliah TI/Antara 2024:2025/belajar-nodejs-dasar-main/web-api.js:3
const request = new XMLHttpRequest();
                  ^
```

ReferenceError: XMLHttpRequest is not defined

```
    at Object.<anonymous> (/Users/asroni/Documents/Kuliah TI/Antara 2024:2025/belajar-nodejs-dasar-main/web-api.js:3:17)
    at Module._compile (node:internal/modules/cjs/loader:1730:14)
    at Object..js (node:internal/modules/cjs/loader:1895:10)
    at Module.load (node:internal/modules/cjs/loader:1465:32)
    at Function._load (node:internal/modules/cjs/loader:1282:12)
    at TracingChannel.traceSync (node:diagnostics_channel:322:14)
    at wrapModuleLoad (node:internal/modules/cjs/loader:235:24)
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:171:5)
    at node:internal/main/run_main_module:36:49
```

Node.js v22.17.1

```
○ asroni@asronis-MacBook-Air belajar-nodejs-dasar-main %
```

Modules



Modules

- Standard Library yang terdapat di NodeJS bisa kita gunakan seperti layaknya JavaScript Modules
- Jika belum mengerti tentang JavaScript Modules, silahkan pelajari kelas saya tentang JavaScript Modules
- Karena NodeJS menggunakan Modules, jika kita ingin menggunakan Modules, kita juga perlu memberi tahu bahwa file JavaScript kita menggunakan Modules, caranya dengan mengubah nama file dari .js menjadi .mjs



Kode : Contoh Standard Library

```
standard-library.mjs x
1 import os from "os"
2
3 console.info(os.platform());
4 console.table(os.cpus());
5
```

Require Function



Require Function

- Awal ketika NodeJS rilis, fitur JavaScript Modules belum rilis, namun sekarang JavaScript sudah banyak menggunakan JavaScript Modules
- NodeJS pun awalnya tidak menggunakan JavaScript Modules, namun sekarang NodeJS sudah bisa menggunakan JavaScript Modules, dan sangat direkomendasikan menggunakannya
- Namun awal sebelum Modules, NodeJS menggunakan function `require()` untuk melakukan import file
- Di materi ini saya sengaja bahas, agar tidak bingung ketika kita melihat tutorial yang masih menggunakan function `require`



Kode : Function Require

```
require-function.js ×  
1  const os = require("os");  
2  
3  console.info(os.platform());  
4  console.table(os.cpus());  
5
```

Global Async di Module



Global Async

- Saat kita belajar JavaScript, untuk menggunakan Async Await, biasanya kita perlu membuat terlebih dahulu function yang kita tandai sebagai async
- Saat kita menggunakan Module, secara default, global code adalah Async, oleh karena itu kita bisa menggunakan Async Await
- Kecuali jika kita membuat function, maka function tersebut harus kita tandai sebagai Async jika ingin menggunakan Async Await



Kode : JavaScript

```
async.js x
1 function samplePromise() {
2     return Promise.resolve( value: "Eko");
3 }
4
5 const data = await samplePromise(); // error
6 console.info(data);
7
```



Kode : JavaScript Module

```
async.mjs x
1 function samplePromise() {
2     return Promise.resolve( value: "Eko");
3 }
4
5 const data = await samplePromise();
6 console.info(data);
7
```

OS



OS

- OS merupakan standard library yang bisa digunakan untuk mendapatkan informasi tentang sistem operasi yang digunakan
- <https://nodejs.org/docs/latest-v22.x/api/os.html>
- <https://nodejs.org/dist/latest-v16.x/docs/api/os.html>



Kode : OS

```
JS os.mjs x
1  import os from "os";
2
3  console.info(os.platform());
4  console.info(os.arch());
5  console.table(os.cpus());
6  console.info(os.uptime());
7  console.info(os.totalmem());
8  console.info(os.freemem());
9  console.table(os.networkInterfaces());
10
```

—

Path



Path

- Path merupakan standard library yang bisa kita gunakan untuk bekerja dengan lokasi file dan directory / folder
- <https://nodejs.org/docs/latest-v22.x/api/path.html>
- <https://nodejs.org/dist/latest-v16.x/docs/api/path.html>

Kode : Path

```
path.mjs x
1  import path from "path";
2
3  const file = "/Users/khannedy/contoh.html";
4
5  console.info(path.sep);
6  console.info(path.dirname(file));
7  console.info(path.basename(file));
8  console.info(path.extname(file));
9  console.info(path.parse(file));
10
```

File System



File System

- File System merupakan standard library yang bisa digunakan untuk memanipulasi file system
- Dalam File System, terdapat 3 jenis library
- Pertama library yang bersifat blocking atau synchronous
- Kedua library yang bersifat non-blocking atau asynchronous menggunakan callback
- Ketika library yang bersifat non-blocking atau asynchronous tapi menggunakan promise
- <https://nodejs.org/docs/latest-v22.x/api/fs.html>
- <https://nodejs.org/dist/latest-v16.x/docs/api/fs.html>



Kode : File System

```
file-system.mjs x
1  import fs from "fs";
2
3  const buffer = fs.readFileSync(path: "file-system.mjs");
4
5  console.info(buffer.toString());
6
7  fs.writeFileSync(file: "temp.txt", data: "Hello World");
8
```



Debugger



Debugger

- NodeJS memiliki fitur debugger, dimana kita bisa mengikuti tahapan eksekusi program di NodeJS
- Hal ini sangat cocok ketika kita melakukan proses debugging, mencari sebab masalah yang terjadi di aplikasi kita
- <https://nodejs.org/docs/latest-v22.x/api/debugger.html>
- <https://nodejs.org/dist/latest-v16.x/docs/api/debugger.html>



Breakpoint

- Dalam debugging, terdapat istilah breakpoint, yaitu lokasi dimana kita ingin menghentikan sementara eksekusi kode program
- Biasanya ini dilakukan untuk mengawasi data-data di sekitar lokasi berhentinya tersebut
- Untuk menambahkan breakpoint, kita bisa menggunakan kata kunci: debugger



Menjalankan Mode Debug

- Jika kita menjalankan file JavaScript hanya dengan menggunakan perintah `node namafile.js`, maka secara default dia tidak akan jalan dalam mode debug
- Agar jalan dalam mode debug, kita harus menambahkan perintah `inspect` :
`node inspect namafile.js`



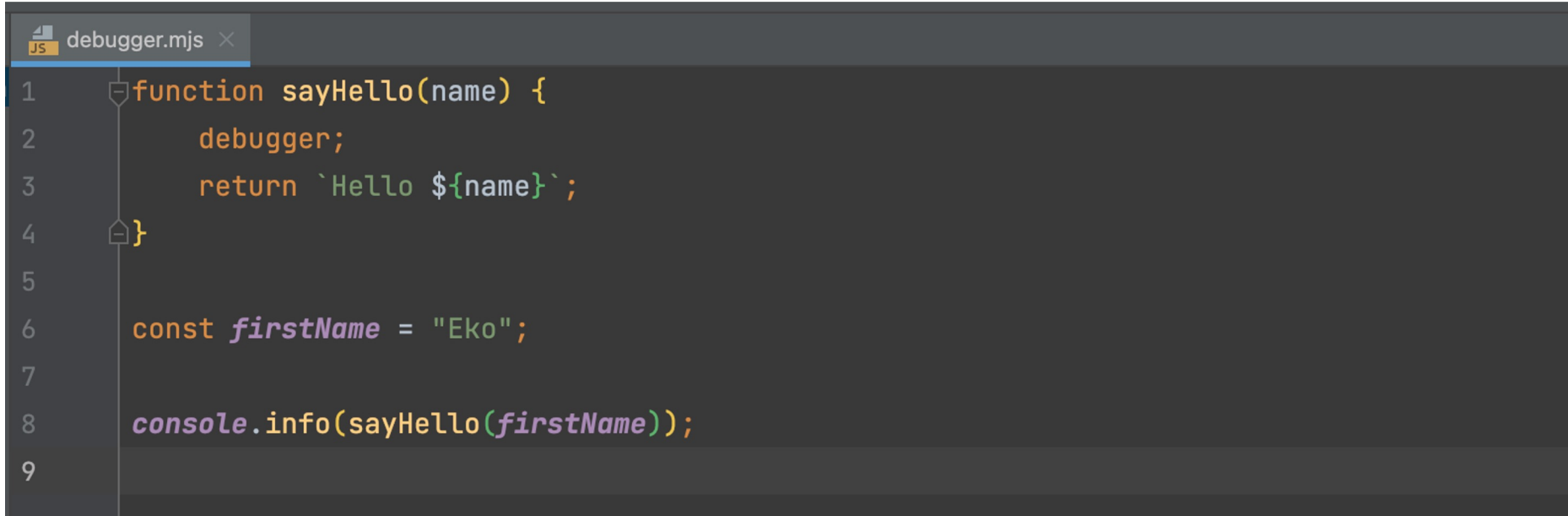
Perintah Debugger

Saat masuk ke mode debug, ada beberapa perintah yang bisa kita gunakan dalam melakukan debugging

- cont, c: Continue execution
- next, n: Step next
- step, s: Step in
- out, o: Step out
- pause: Pause running code



Kode : Debugger



```
1 function sayHello(name) {  
2     debugger;  
3     return `Hello ${name}`;  
4 }  
5  
6 const firstName = "Eko";  
7  
8 console.info(sayHello(firstName));  
9
```

DNS



DNS

- DNS merupakan standard library yang bisa digunakan untuk bekerja dengan DNS (domain name server)
- <https://nodejs.org/dist/latest-v16.x/docs/api/dns.html>

Kode : DNS

```
dns.mjs x
1  import dns from "dns";
2
3  function callback(error, ip) {
4      console.info(ip);
5  }
6
7  dns.lookup( hostname: "www.programmerzamannow.com", callback)
8
```



Kode : DNS Promise

```
JS dns.mjs x
1 import dns from "dns/promises";
2
3 const lookup = await dns.lookup( hostname: "www.programmerzamannow.com");
4
5 console.info(lookup.family);
6 console.info(lookup.address);
7
```