WEB PROGRAMMING

MODUL PRAKTIKUM

DAFTAR ISI

[8 Node.js 2](#_Toc54476273)

[8.1 Pendahuluan 2](#_Toc54476274)

[8.1.1 Definisi 2](#_Toc54476275)

[8.1.2 Instalasi 3](#_Toc54476276)

[8.1.3 Visual Studio Code Extension 4](#_Toc54476277)

[8.1.4 Hello World 4](#_Toc54476278)

[8.1.5 Default Entry 5](#_Toc54476279)

[8.1.6 Hello World Server 6](#_Toc54476280)

[8.1.7 NPM Node Package Manager 7](#_Toc54476281)

[8.2 Callback 9](#_Toc54476282)

[8.2.1 Event 9](#_Toc54476283)

[8.2.2 Event Emitter 10](#_Toc54476284)

[8.2.3 Asynschronous Callback 11](#_Toc54476285)

[8.3 File System 12](#_Toc54476286)

[8.3.1 Synchronous vs Asynchronous 12](#_Toc54476287)

[8.3.2 Open 13](#_Toc54476288)

[8.3.3 Write 14](#_Toc54476289)

[8.3.4 Read 15](#_Toc54476290)

[8.3.5 Close 16](#_Toc54476291)

[8.3.6 Delete 17](#_Toc54476292)

# Node.js

## Pendahuluan

### Definisi

Node.js adaalah sebuah Runtime yang terpasang di Web Server untuk menangani Request dari client seperti Web Browser dan Mobile App, dan menggunakan JavaScriptEngine untuk mentranslate JavaScript menjadi bahasa mesin. Node.js bukanlah sebuah bahasa pemrograman, tetapi menggunakan Java Script sebagai bahasa antar muka. Node.js juga menyediakan pustaka yang kaya dari berbagai modul JavaScript yang menyederhanakan pengembangan aplikasi web menggunakan Node.js secara luas.

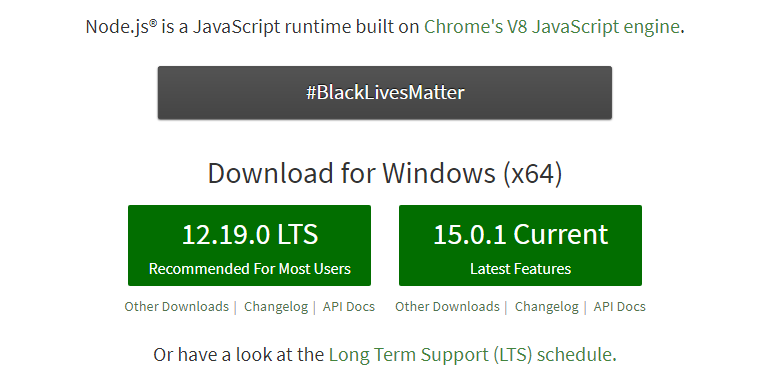
Node.js memiliki sifat Non-Blocking Asynchronous, artinya pada satu saat dapat menangani beberapa request sekaligus. Berbeda dengan sifat Syncronous dimana sebuah proses harus diselesaikan terlebih dahulu sebelum menangani proses lain. Dikarenakan sifat ini, Node.js sangat scalable, yaitu dapat dengan mudah menambah kapasitas proses hanya dengan menambah server.

Diagram di bawah ini menggambarkan module Node.js yang sudah tersedia. Pengembang dapat menambahkan sendiri modulenya apabila diperlukan.

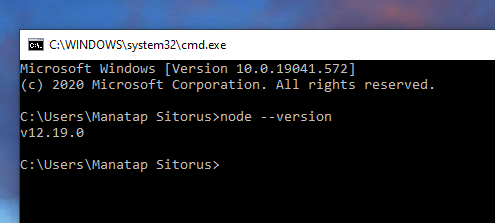


### Instalasi

Kunjungi situr Node.js dengan Url: <https://nodejs.org> lalu unduh versi LTS (Long Term Support). Versi LTS lebih banyak digunakan dan sudah banyak bantuan apabila menghadapi suatu masalah. Fitur-fitur baru ada di versi Current, tetapi masih ada kemungkinan bug.

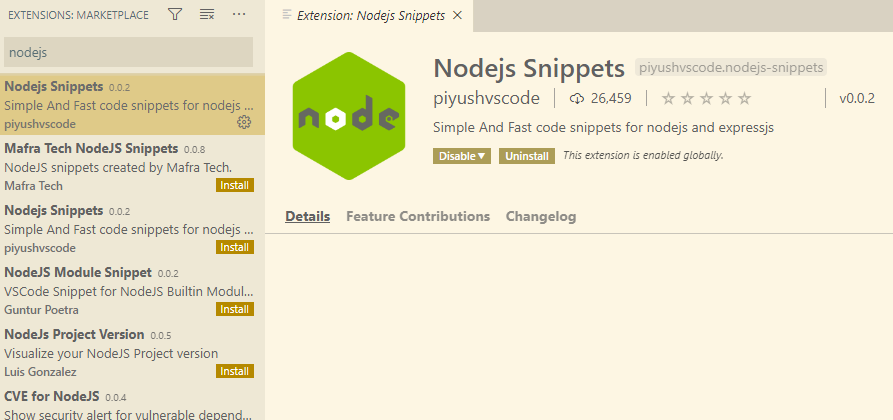


Unduh, lalu ikuti panduan instalasinya. Centang pilihan `Automatically install the necessary components…`. Setelah selesai melakukan instalasi, bukalah command prompt, lalu cari tahu versi Node.js menggunakan perintah di bawah ini.



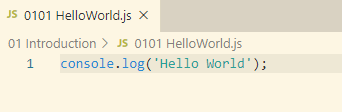
### Visual Studio Code Extension

Marketplace di Visual Studio Code banyak memberikan extension utk Node.js snippets. Salahs atunya adalah Modejs Snippets. Install extension ini apabila diperlukan.

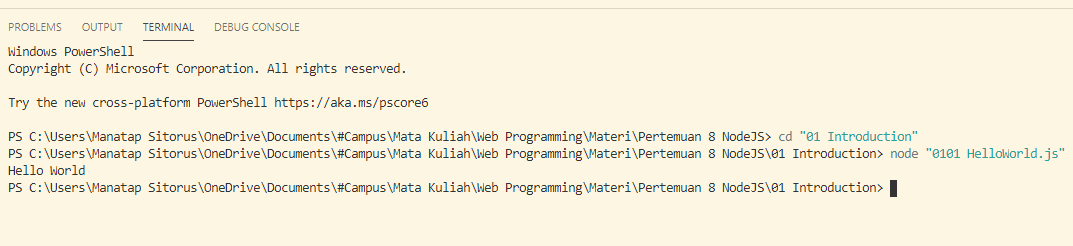


### Hello World

Gunakan Visual Studio Code untuk mulai membuat aplikasi pertama. Boleh menggunakan code editor lain. Pertama siapkan folder sebagai tempat untuk menyiapkan semua file JS.

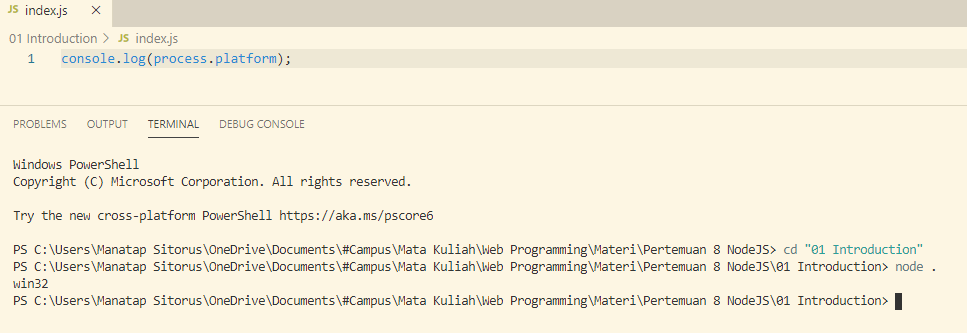


Simpan lalu panggil Terminal (Menu Terminal – New Terminal). Posisikan terminal supaya berada di direktori yang sama dengan file 0101 HelloWorld.js, gunakan perintah cd “Nama Direktori”. Setelah berada di direktori yang sama, ketikkan node “0101 HelloWorld.js” lalu Terminal akan memberikan teks Hello World.



### Default Entry

Selain memanggil dengan nama file seperti contoh sebelumnya, dapat juga dipanggil dengan simbol . (titik). Perintah node . memanggil file Index.js.



### Hello World Server

Di contoh-contoh sebelumnya semua proses dilakukan di console, dan belum benar-benar berjalan untuk web. Pada contoh berikut sudah menggunakan module http untuk membuat web server sederhana. Setelah selesai membuat, jalankan di Terminal dengan perintah node “0103 HelloWorldServer.js”. Kemudian panggil browser lalu ketikkan ip lokal dengan port 8081 sesuai yang tertera di coding.



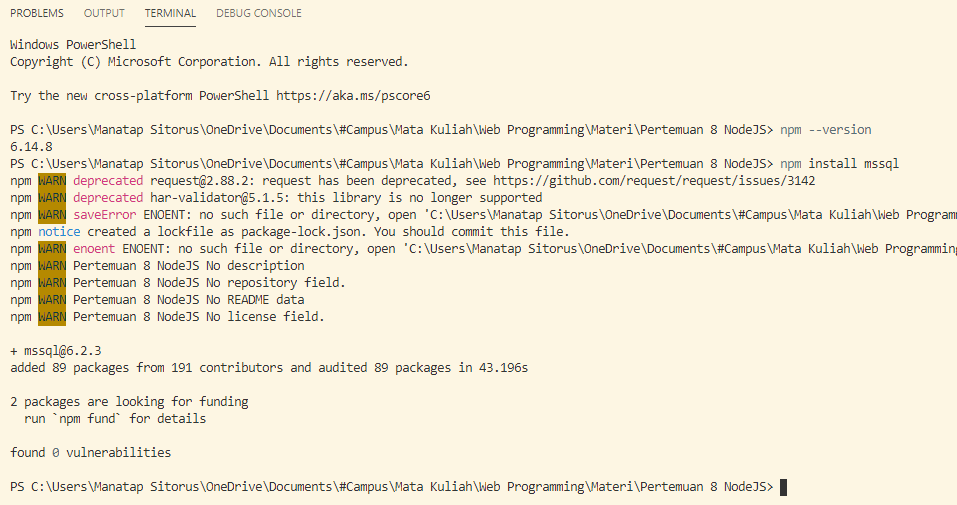
Selama command di Terminal belum di-terminate, maka Web Server akan tetap running menunggu request dari browser. Untuk menghentikan proses, tekan Ctrl+C di Terminal. Lalu coba panggil kembali Url yang sama, seharusnya akan menghasilkan error. Hal ini dikarenaka Web Server sudah tidak lagi running.

### NPM Node Package Manager

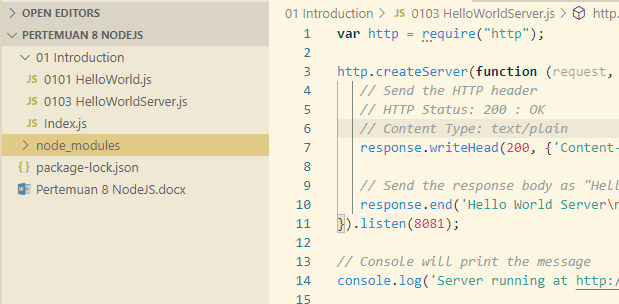
Node Package Manager (NPM) menyediakan dua fungsi utama:

* Repositori online untuk package / modul Node.js yang dapat dicari di www.npmjs.com
* Command Utility untuk menginstal package Node.js, melakukan manajemen versi dan dependency management Node.js package.

Untuk melihat versi npm yang terinstall gunakan perintah npm --version. NPM secara default terinstall bersama installer Node.js Untuk melakukan instalasi sebuah paket gunakan perintah seperti berikut: npm install <PackageName>, contoh untuk menginstall paket konektivitas ke SQL Server gunakan npm install mssql.



Secara default, NPM menginstal dependensi dalam mode lokal. Mode lokal mengacu pada instalasi paket di direktori node\_modules yang terletak di folder tempat aplikasi Node. Paket yang digunakan secara lokal dapat diakses melalui metode require(). Untuk melakukan instalasi secara global, gunakan perintah npm install <PackageName> -g.



Berikut adalah daftar beberapa perintah NPM yang dapat digunakan:

* Melihat daftar packacge yang sudah diinstall: npm ls
* Menghapus package: npm uninstall mssql
* Mengupdate package: npm update mssql

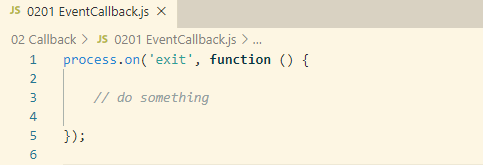
## Callback

Callback adalah fungsi asinkron (Asyncronous Function). Fungsi callback dipanggil setelah menyelesaikan tugas tertentu. Node menggunakan banyak callback. Semua API dari Node ditulis sedemikian rupa sehingga mendukung callback.

Misalnya, fungsi untuk membaca file dapat mulai membaca dan mengembalikan kontrol ke lingkungan eksekusi segera sehingga instruksi selanjutnya dapat dijalankan. Setelah file I / O selesai, akan memanggil fungsi callback sambil meneruskan fungsi callback, konten file sebagai parameter. Jadi tidak ada pemblokiran atau tunggu File I / O. Hal ini membuat Node.js sangat skalabel, karena dapat memproses permintaan dalam jumlah besar tanpa menunggu fungsi apa pun memberikan hasil.

### Event

Event dan Callback selalu berjalan bersamaan. Event adalah suatu kondisi yang diterima oleh sebuah Handler. Saat suatu event terjadi, maka Node akan memanggil sebuah fungsi callback.



### Event Emitter

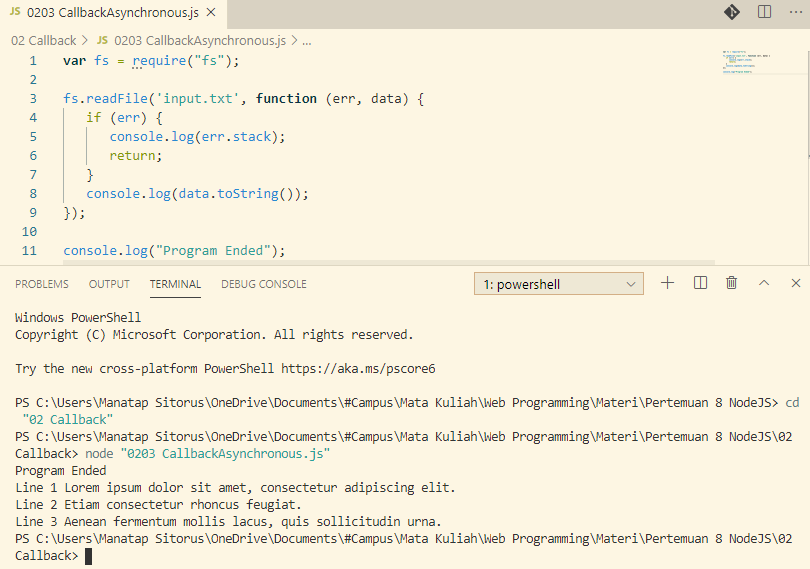
Seperti di banyak pemrograman lainnya, event di Node dapat dibentuk oleh programmer. Selain event default yang dimiliki di module node.js kita bisa membuat event sesuai kebutuhan aplikasi.

Tahap awal gunakan module events lalu buatlah variabel Emitter. Lalu daftarkan nama event yang diinginkan, lalu panggil event tersebut dengan nama event yang didaftarkan sebelumnya.



### Asynschronous Callback

Dalam Aplikasi Node, setiap fungsi asinkron menerima callback sebagai parameter terakhir dan fungsi callback menerima kesalahan sebagai parameter pertama. Contoh di bawah ini menunjukkan sebuah fungsi Callback yang baru dijalankan setelah keseluruhan fungsi selesai dijalankan.

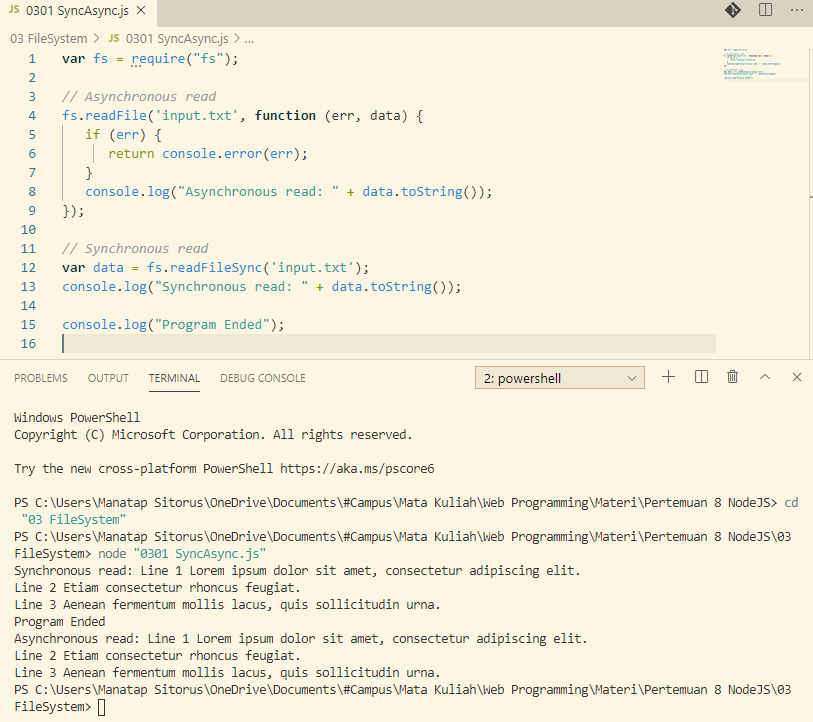


## File System

Node mengimplementasikan File I / O menggunakan wrapper sederhana di fungsi POSIX standar. Modul Sistem File Node (fs) dapat diimpor menggunakan sintaks var fs = require("fs").

### Synchronous vs Asynchronous

Setiap metode dalam modul fs memiliki bentuk sinkron dan asinkron. Metode asinkron mengambil parameter terakhir sebagai callback fungsi penyelesaian dan parameter pertama dari fungsi callback sebagai kesalahan. Metode asinkron sebaiknya digunakan daripada metode sinkron, karena metode asinkron tidak memblokir program selama eksekusinya, sedangkan metode sinkron memblokir alur program selanjutnya.



### Open

Untuk membuka sebuah file untuk ditulis atau dibaca, gunakan perintah berikut

fs.open(path, flags[, mode], callback)

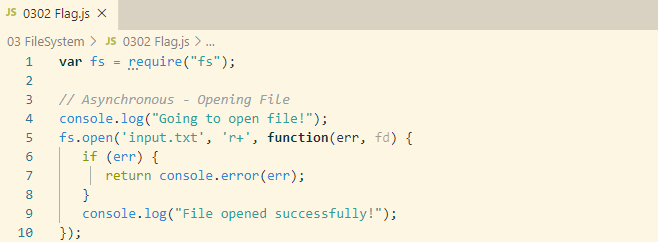
Berikut adalah deskripsi parameter:

* Path - string nama file termasuk path.
* Flag - menunjukkan perilaku file yang akan dibuka.
* Mode - mengatur mode file (permission dan sticky bits), tetapi hanya jika file dibuat. Defaultnya adalah 0666, dapat dibaca dan ditulisi.
* Callback - fungsi callback yang mendapat dua argumen (err, fd).

Berikut adalah tipe Flag:

|  |  |
| --- | --- |
| r | Membuka file untuk dibaca |
| r+ | Membuka file untuk dibaca dan ditulis |
| rs | Membuka file dalam mode Sinkron |
| rs+ | Membuka file untuk dibaca dan ditulis dalam mode Sinkron |
| w | Membuka file untuk ditulis. File akan dibentuk apabila tidak ada, atau dihapus terlebih dahulu apabila sudah ada konten. |
| wx | Seperti w tetapi akan gagal apabila path sudah ada |
| wx+ | Seperti w+ tetapi akan gagal apabila path sudah ada |
| a | Membukan file untuk dilanjutkan (append). File akan dibentuk apabila tidak ada |
| ax | Seperti a tetapi akan gagal apabila path sudah ada |
| a+ | Membuka file untuk dibaca dan dilanjutkan (append). File akan dibentuk apabila tidak ada |
| ax+ | Seperti a+ tetapi akan gagal apabila path sudah ada |

Contoh penggunakan tipe Flag dalam fungsi Open.



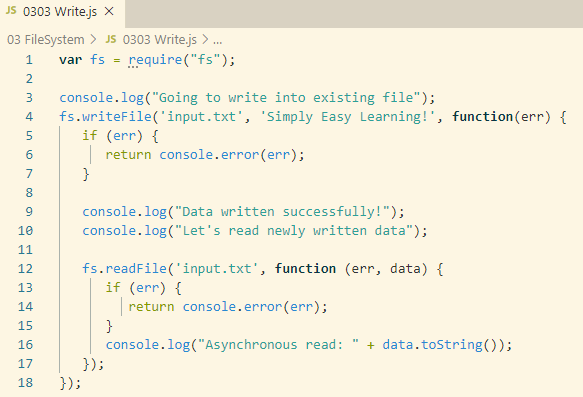
### Write

Untuk menulis sebuah file, gunakan perintah berikut

fs.writeFile(filename, data[, options], callback)

Berikut adalah deskripsi parameter:

* path - string nama file termasuk path.
* data - String atau Buffer yang akan ditulis ke dalam file.
* options - {encoding, mode, flag}. Secara default pengkodeannya adalah utf8, mode adalah nilai oktal 0666. dan flag adalah 'w'
* callback - fungsi panggilan balik yang memiliki 1 parameter error untuk mengembalikan pesan jika ada error saat penulisan



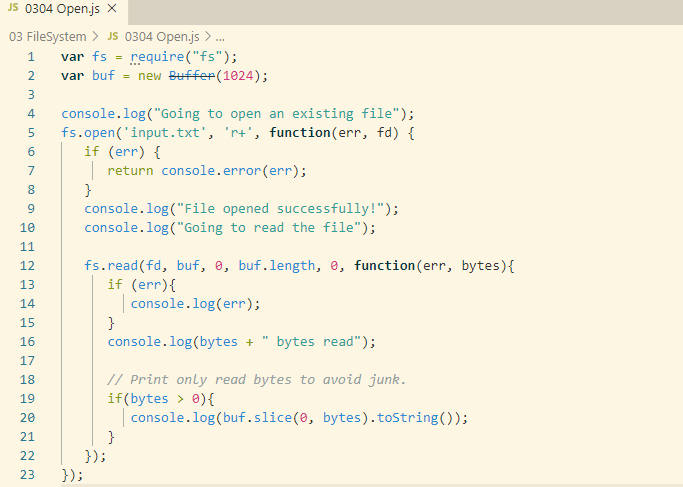
### Read

Untuk menulis sebuah file, gunakan perintah berikut

fs.read(fd, buffer, offset, length, position, callback)

Berikut adalah deskripsi parameter:

* fd - deskriptor file yang dikembalikan oleh fs.open ().
* buffer - tempat data akan ditulis.
* offset - offset dalam buffer untuk mulai menulis.
* length - bilangan bulat yang menentukan jumlah byte untuk dibaca.
* position - bilangan bulat yang menentukan dari mana harus mulai membaca dalam file. Jika posisi nol, data akan dibaca dari posisi file saat ini.
* callback - fungsi callback yang mendapatkan tiga argumen, (err, bytesRead, buffer).



### Close

Untuk menulis sebuah file, gunakan perintah berikut

fs.close(fd, callback)

Berikut adalah deskripsi parameter:

* fd - deskriptor file yang dikembalikan oleh fs.open ().
* callback - fungsi callback yang aka dijalankan saat menutup file



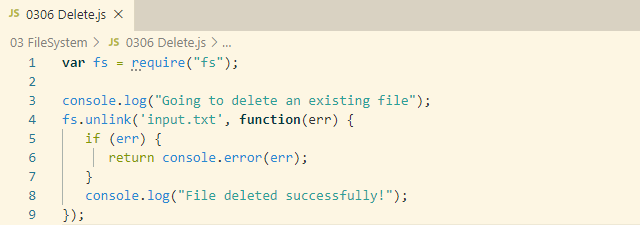
### Delete

Untuk menulis sebuah file, gunakan perintah berikut

fs.unlink(path, callback)

Berikut adalah deskripsi parameter:

* path - nama file termasuk path.
* callback - fungsi callback



## Web Server

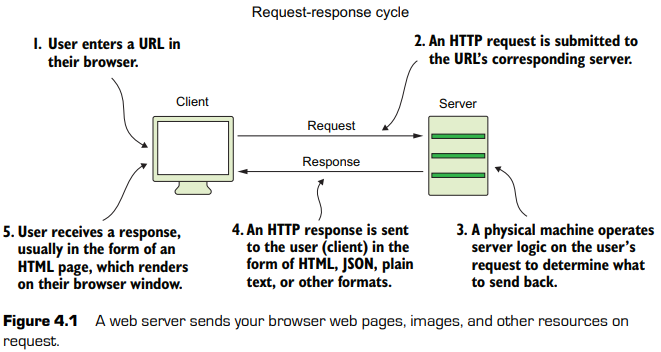
Web Server adalah perangkat lunak yang dirancang untuk menanggapi permintaan melalui internet dengan memuat atau memproses data. Pikirkan server web seperti teller bank, yang tugasnya memproses permintaan Anda untuk menyetor, menarik, atau melihat uang di akun Anda.

Sama seperti teller bank mengikuti protokol untuk memastikan bahwa mereka memproses permintaan Anda dengan benar, server web mengikuti Hypertext Transfer Protocol (HTTP), sistem standar yang diamati secara global untuk melihat halaman web dan mengirim data melalui internet. Salah satu cara klien (komputer Anda) dan server berkomunikasi adalah melalui HTTP Verb.

Verb ini menunjukkan jenis permintaan yang dibuat, seperti apakah pengguna mencoba memuat halaman web baru atau memperbarui informasi di halaman profil mereka. Konteks interaksi pengguna dengan aplikasi adalah bagian penting dari respons permintaan.

Berikut dua metode HTTP yang paling banyak digunakan yang akan Anda temui:

* GET — Metode ini meminta informasi dari server. Biasanya, server merespons dengan konten yang bisa Anda lihat kembali di browser Anda (seperti dengan mengklik link untuk melihat halaman beranda situs).
* POST — Metode ini mengirimkan informasi ke server. Server mungkin merespons dengan halaman HTML atau mengarahkan Anda ke halaman lain dalam aplikasi setelah memproses data Anda (seperti mengisi dan mengirimkan formulir pendaftaran).



### Simple Web Server

Line 3. Untuk memulai membuat web server, pertama tentukan port yang digunakan. Port 3000 sering digunakan untuk development.

Line 5. Import module http dan http-status-codes. Module http digunakan untuk membuat server. Module http-status-codes untuk mengembalikan statu response.

Line 7. Buatlah server yg menerima argumen request dan response, kemudian mengembalikan teks yang akan dimunculkan di browser client.

Line 10. Siapkan reponse yang akan diterima oleh browser client. Pertama, siapkan Header.

Line 12. Teks yg akan diterima oleh client.

Line 14. Tuliskan body response sesuai teks yang akan diterima client.

Line 16. Tutup response, agar segera dikirim ke browser client.

Line 21. Web Server diminta untuk menerima request di port 3000. Apabila dicoba menggunakan port lain, maka tidak akan ada response.



Pada saat menjalankan aplikasi ini dengan perintah node “0401 WebServer.js” akan ada kemungkinan mengembalikan error. Periksa module apa yang harus ditambahkan, lalu jalankan dengan perintah npm install.

### Handling Request

Di source code sebelumnya terdapat objek yang memiliki fungsi callback (req, res) ⇒ {}. Fungsi ini dijalankan setiap kali permintaan dibuat ke server. Jika client mengunjungi localhost: 3000, kemudian me-refresh halaman, fungsi callback dijalankan dua kali, pertama saat load, kedua saat refresh.

Source code di bawah ini memisahkan fungsi membuat server dan response saat client memanggil web server. Perhatikan di line 6 terdapat event request yang memanggil callback. Fungsi tersebut mengembalikan teks ke browser.



Setelah selesai menggunakan Web Server, stop dengan perintah Ctrl+C di Terminal Visual Studio Code.

### Handling Request With Data

Seringkali sebuah request yang dikirim dari client memiliki data. Dengan menggunakan source code sebelumnya, tambahkan event data dan end.

Line 14. Request diterima, kemudian dimasukkan ke dalam array Body.

Line 18. Request Body yang sudah diterima, ditampilkan ke console.

Line 23 s/d 25. Menampilkan informasi Request dalam bentuk JSON.

Line 6. Fungsi untuk mengubah object menjadi JSON, gunakan perintah JSON.stringify().

Line 31. Menampilkan pesan ke client.

Untuk mensimulasikan request dan data, gunakan curl. Panggil Windows PowerShell lalu jalankan dengan user Administrator. Cek dahulu menggunakan perintah curl --version, apabila curl sudah terinstall maka akan ada pesan mengenai versi curl. Apabila belum, install menggunakan Chocolatey, perintahnya adalah choco install curl.

Apabila Chocolatey belum terinstall, kunjungi url <https://chocolatey.org/install> lalu ikuti langkah-langkah di tab Individual. Instalasi node.js yang lengkap secara otomatis akan menginstall Chocolatey.

Setelah requirement di atas sudah terpenuhi, jalankan source code dibawah dengan perintah node “0403 HandlingRequestWithData.js”, dengan begini Web Server sudah running.

Langkah selanjutlan memanggil web server tersebut dari Windows PowerShell. Jalankan perintah curl --data “username=namasaya&password=rahasia” <http://127.0.0.1:3000>.

Apabila ada pesan error Invoke-WebRequest : A positional parameter cannot be found that accepts argument 'usernam… artinya curl yang sudah terinstall bentrok dengan curl original windows 10. Jalankan perintah Remove-item alias:curl di PowerShell. Tidak akan ada pesan apapun setelah perintah tersebut dijalankan, tetapi seharusnya sudah sukses. Setelah menghapus alias tersebut, jalankan kembali perintah curl --data.

Setelah sukses dijalankan maka akan ada pesan <h1>This will show on the screen.</h1> di PowerShell. Cek di Terminal Visual Studio Code, maka console sudah terisi dengan informasi body, method, url, headers.

Coba jalankan di browser dengan url <http://127.0.0.1:3000/?username=namasaya&password=rahasia> cek perbedannya.

Jalankan pula di aplikasi RESTClient seperti Postman atau Insomnia, lalu lihat response dan console di terminal.

Setelah selesai menggunakan Web Server, stop dengan perintah Ctrl+C di Terminal Visual Studio Code.



## Routing

Rute adalah cara untuk menentukan bagaimana aplikasi harus menanggapi permintaan yang dibuat ke URL tertentu. Aplikasi harus mengarahkan permintaan ke beranda atau login tergantung dari permintaan client, contoh: halo.com, halo.com/info, halo.com/login, halo.com/signup

### Simple Routing

Line 1. Routing sederhana berupa rangkaian dictionary yang terdiri dari Key dan Value. Di contoh ini setiap value hanya berupa html simpel. Di contoh yang belih besar berupa full page.

Line 19. Dengan menggunakan dictionary yang kita punya, panggil value sesuai URL yang direquest oleh client.



### Serving Static Files With Prepared Dictionary

Untuk memulai menggunakan reponse full page setiap kali ada request dari client, tambahkan sebuah folder bernama Views, lalu buat file index.html. File ini akan dipanggil saat menerima request.

Line 6. Sebelumnya menggunakan dictionary yang value-nya berisi html text. Sekarang berisi alamat dari sebuah file.

Line 16. Fungsi readFile digunakan untuk membaca file index lalu mengembalikan text sebagai response untuk client.



### Serving Static Files With Dynamic View

Siapkan beberapa file di folder views misalnya SignIn.html, SignUp.html, About.html.

Line 6. Fungsi ini digunakan untuk memanggil web page sesuai url dari client.

Gunakan url http://127.0.0.1:3000/SignUp atau lainnya untuk melihat bagaimana interaksi antara url dan file yang ada di folder Views.



### Moving Router Into Another Files

Seiring bertambahnya file utama, kemampuan untuk memfilter semua kode yang ditulis menjadi lebih rumit. Untuk mengatasi masalah ini, pindahkan rute ke file baru bernama router.js. Juga atur ulang cara menyimpan dan menangani rute.

Sebuah file Router.js biasanya terdiri dari Get Post dan Handle.



