

Node.JS

Introduction to Node.JS



NodeJS Introduction

- ☐ Knowing NodeJS
- ☐ Runtime Environment
- ☐ NodeJS Module
- ☐ NodeJS Package Manager
- ☐ NodeJS Read and Write File

NodeJS Introduction

- Mengetahui tentang NodeJS
- Mengetahui tentang runtime environment
- Mengetahui tentang module di NodeJS
- Mengetahui tentang package manager di NodeJS
- Mengetahui tentang read dan write di NodeJS

Objektif Sesi

- Student memahami tentang NodeJS
- Student memahami tentang runtime environment
- Student memahami module di NodeJS
- Student memahami package manager di NodeJS
- Student memahami read dan write file di NodeJS

NodeJS Introduction



Knowing NodeJS



Runtime Environment



NodeJS Module

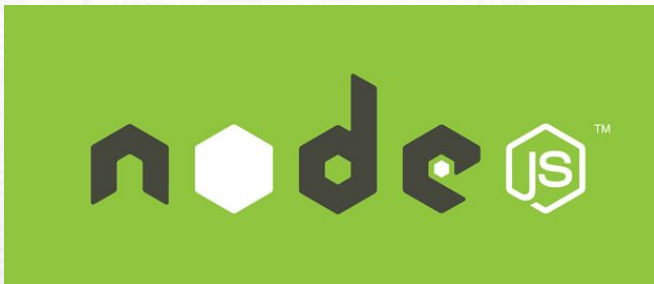


NodeJS Package Manager



NodeJS Read and Write File

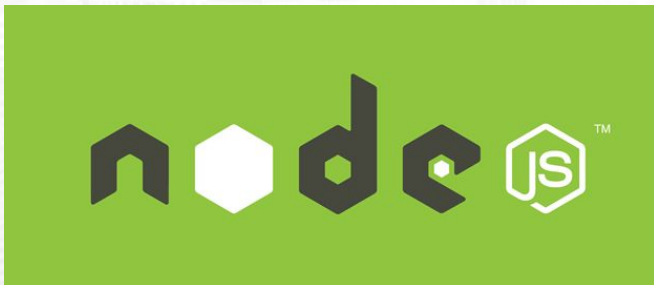
Knowing NodeJS



Node.js adalah runtime dari JavaScript yang memiliki kemampuan bisa melakukan running aplikasi Javascript diluar browser. Ryan Dahl adalah creator yang mengembangkan Node di tahun 2009, dan release pertamanya, dan ada versi 15.14, dirilis pada April 2021.

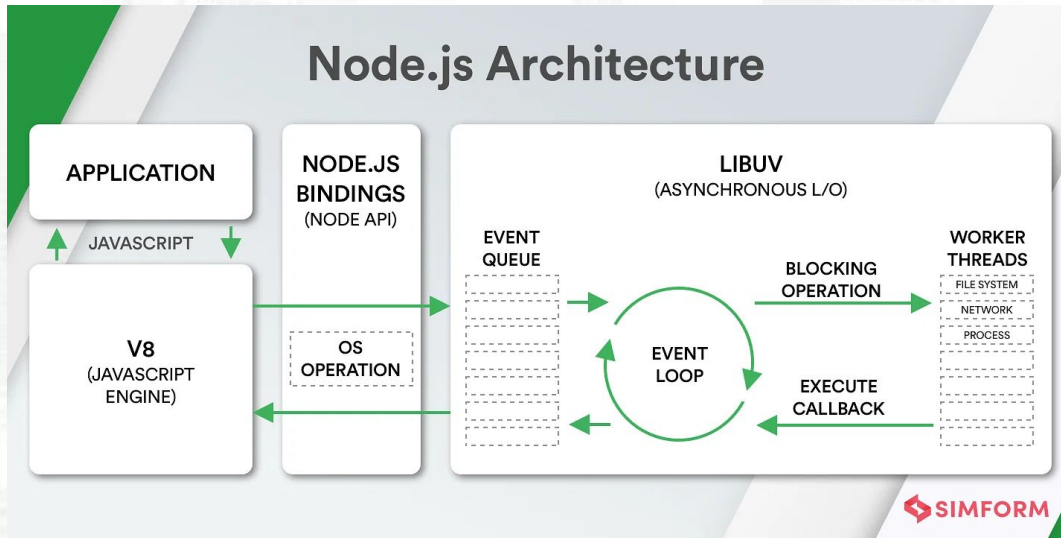
Pengembang menggunakan Node.js untuk membuat aplikasi web sisi server, dan sangat cocok untuk aplikasi intensif data karena model asynchronous blocking dari Node

Why we use NodeJS?



- NodeJS dibangun di mesin V8 Google Chrome, dengan ini membuat aplikasi Node menjadi sangat cepat
- Karena NodeJS tidak perlu menunggu proses selesai duluan karena model asynchronous yang dimiliki NodeJS membuat eksekusi aplikasi menjadi lebih non blocking atau paralel
- Karena NodeJS adalah open-source dan tidak lain adalah kerangka kerja JavaScript

Architecture of NodeJS?



Disamping adalah ilustrasi arsitektur dari NodeJS. Disana terlihat proses dimana aplikasi berbasis Node berinteraksi dengan OS

NodeJS Introduction



Knowing NodeJS



Runtime Environment



NodeJS Module



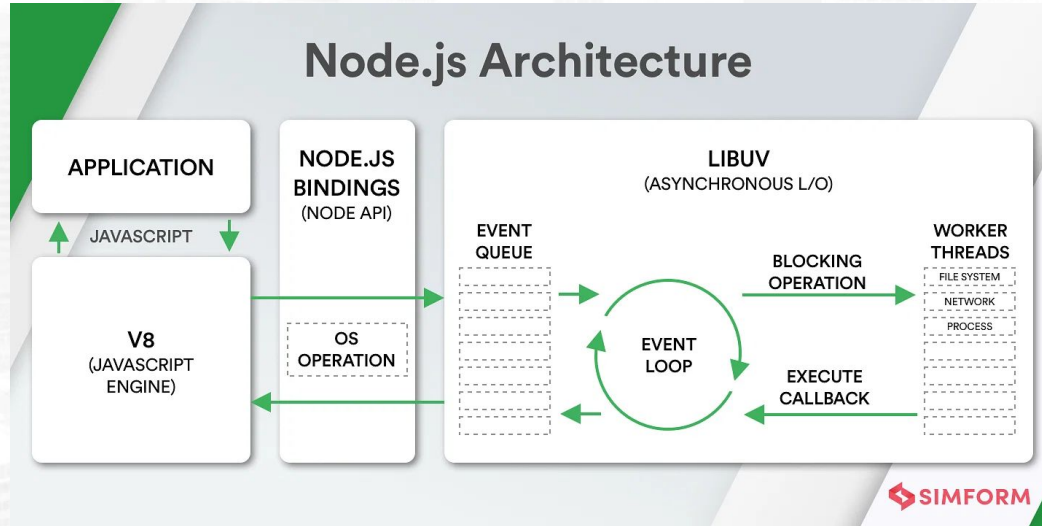
NodeJS Package Manager



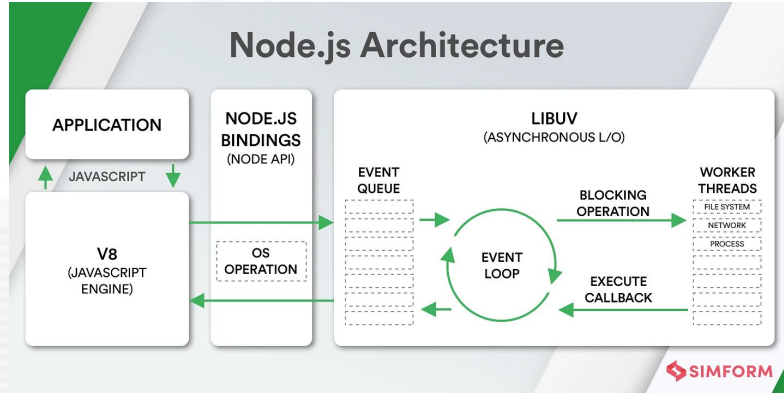
NodeJS Read and Write File

Runtime Environment

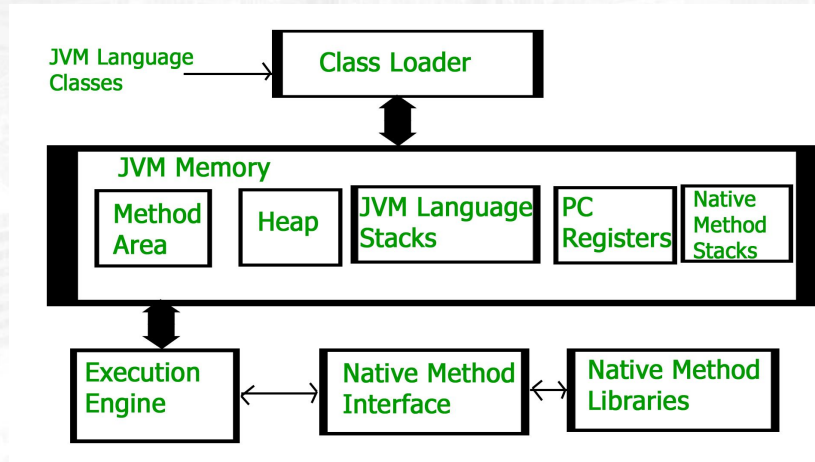
Runtime environment adalah environment di mana program atau aplikasi dijalankan.



Runtime Environment



NodeJS runtime environment seperti JVM di dunia Java programming. Disamping adalah analogi dari runtime Node dan JVM



NodeJS Introduction



Knowing NodeJS



Runtime Environment



NodeJS Module



NodeJS Package Manager



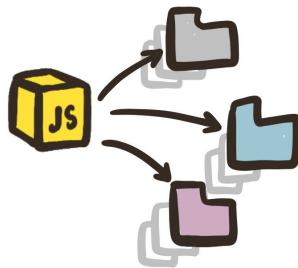
NodeJS Read and Write File

NodeJS Module

THREE WAYS TO SHARE

NODE 

MODULES ACROSS
MULTIPLE PROJECTS



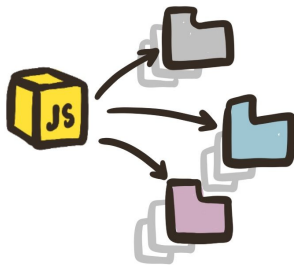
Di Node.js, Module adalah baris kode yang di enkapsulasi yang berkomunikasi dengan aplikasi eksternal berdasarkan fungsi terkait. Alasan dari programmer sangat bergantung pada modul adalah karena kegunaannya bisa reusable.

NodeJS Module Categories

THREE WAYS TO SHARE

NODE 

MODULES ACROSS
MULTIPLE PROJECTS



Modules are of three types:

- Core Modules
- Local Modules
- Third-party Modules

NodeJS Module Categories - Core Module



```
var http = require('http');  
http.createServer(function (req, res) {  
  res.writeHead(200, {'Content-Type': 'text/html'});  
  res.write('Welcome to this page!');  
  res.end();  
}).listen(3000);
```

Node.js memiliki banyak modul bawaan yang merupakan bagian dari platform dan dilengkapi dengan instalasi Node.js.

Dalam contoh di atas, fungsi `require()` mengembalikan objek karena module `http` mengembalikan value berbentuk objek.

NodeJS Module Categories - Local

M

```

exports.add = function (x, y) {
  return x + y;
};

exports.sub = function (x, y) {
  return x - y;
};

exports.mult = function (x, y) {
  return x * y;
};

exports.div = function (x, y) {
  return x / y;
};

```

Tidak seperti modul bawaan dari Node dan eksternal public repository, modul local dibuat secara lokal. Di samping adalah contoh modul penghitungan sederhana yang menghitung berbagai operasi.

Buat file **calc.js** yang memiliki kode berikut:

NodeJS Module Categories - Local Module

```
var calculator = require('./calc');  
  
var x = 50, y = 10;  
  
console.log("Addition of 50 and 10 is "  
           + calculator.add(x, y));  
  
console.log("Subtraction of 50 and 10 is "  
           + calculator.sub(x, y));  
  
console.log("Multiplication of 50 and 10 is "  
           + calculator.mult(x, y));  
  
console.log("Division of 50 and 10 is "  
           + calculator.div(x, y));
```

Karena file ini menyediakan atribut ke dunia luar melalui ekspor, file lain dapat menggunakan fungsi ekspornya menggunakan fungsi `require()`.

Di sini, di file **index.js**

NodeJS Module Categories - Third Party Module



Modul third party adalah modul yang tersedia online menggunakan Node Package Manager (NPM).

Modul-modul ini dapat diinstal di folder project atau secara global di local computer. Beberapa modul third party yang populer adalah mongoose, express, angular, dan react.

NodeJS Introduction



Knowing NodeJS



Runtime Environment



NodeJS Module

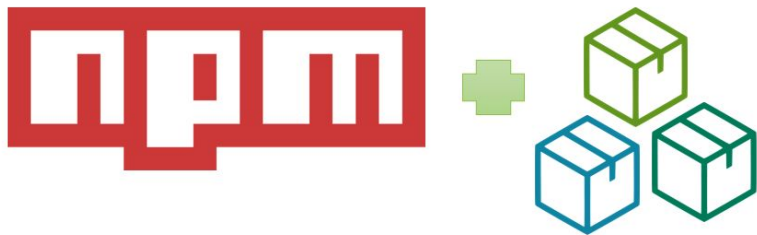


NodeJS Package Manager



NodeJS Read and Write File

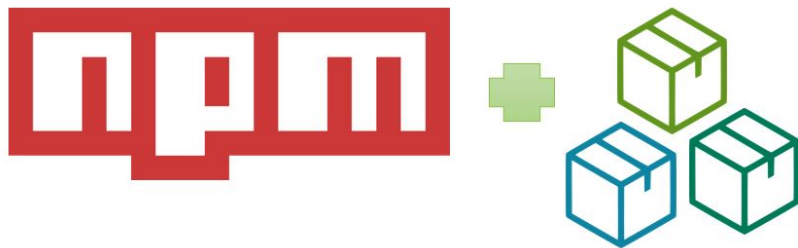
NodeJS Package Manager



Node Package Manager (NPM) merupakan package manager untuk JavaScript yang dapat memudahkan kita dalam mengelola package yang tersedia pada Javascript. NPM ini merupakan standard package manager yang disediakan oleh Node.js dan otomatis terpasang ketika memasang Node.js pada local computer

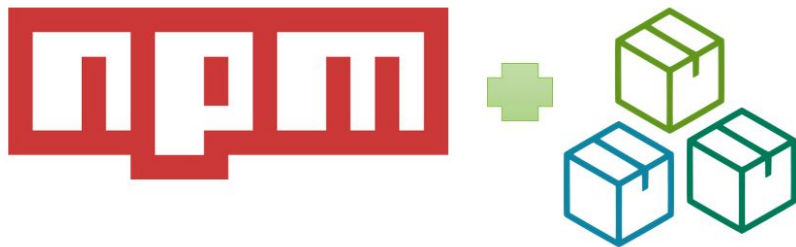
NodeJS Package Manager

- NPM
- Yarn
- PNPM



NodeJS Package Manager

Disini beberapa contoh penerapan package manager di Node



\$ npm install axios

\$ yarn add axios

\$ pnpm install axios

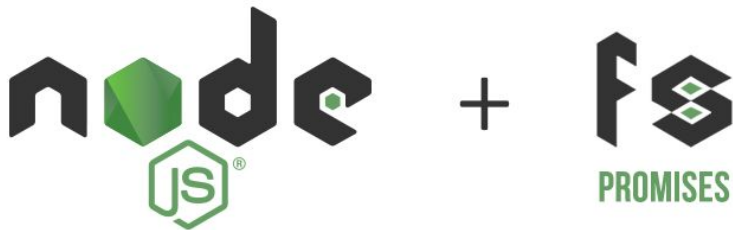
NodeJS Introduction

- ☒ Knowing NodeJS
- ☒ Runtime Environment
- ☒ NodeJS Module
- ☒ NodeJS Package Manager



NodeJS Read and Write File

NodeJS Read and Write File



Node FS adalah module yang mampu membaca dari file di sistem lokal dan bisa sangat berguna. Bisa juga melakukan proses import data dari spreadsheet dan file xml atau apa pun.

NodeJS Read and Write File - Read Files



```
var fs = require("fs");  
  
fs.readFile("temp.txt", "utf-8", (err, data) => {  
  console.log(data);  
});
```

Buat **temp.txt** dalam direktori yang sama. Jalankan kode disamping menggunakan command **node app.js** dan kita akan melihat hasil pembacaan file di console

Berikut contoh untuk membaca file menggunakan Node FS. Setelah menulis kode di bawah ini, Anda dapat menjalankan menggunakan perintah:

\$ node app.js

NodeJS Read and Write File - Write Files



```
var fs = require("fs");

var data = "New File Contents";

fs.writeFile("temp.txt", data, (err) => {
  if (err) console.log(err);
  console.log("Successfully Written to File.");
});
```

Disamping adalah contoh dari penerapan write file di dalam Node FS module.

Berikut contoh untuk menulis file menggunakan Node FS. Setelah menulis kode di bawah ini, Anda dapat menjalankan menggunakan perintah:

\$ node app.js

Study Case

Ada sampel untuk data log sebuah aplikasi [Log sample](#)

Dengan menggunakan Node FS module,

1. Buat file dengan nama **logging.js**
2. Simpan file log diatas dan masukkan data log sampel diatas di dalam file **logging.txt** menggunakan node fs
3. Setelah memasukkan data log kedalam file, baca file log sebelumnya menggunakan node fs

Usahakan hasil kompilasi tidak error ya hehe.

Happy cracking!

Study Case Explained

Dari study case diatas, dibawah ini adalah bagaimana mengerjakan dari study case diatas

1. Teman teman bisa membuka link [berikut](#) untuk melihat data log sample nya
2. Step kedua, teman teman bisa membuat file yang namanya **logging.js**
3. Di dalam file **logging.js**, teman teman bisa mulai menulis syntax untuk melakukan write log sample diatas dalam file **logging.txt**. Referensi kodenya bisa [disini](#)
4. Setelah itu kodenya selesai ditulis, maka teman teman bisa melakukan running file **logging.js** dengan command **node logging.js**
5. Jika kode teman teman benar, maka akan muncul file baru bernama **logging.txt** yang berisi log sample diatas
6. Setelah itu teman teman, bisa membuat kode lagi untuk membaca data dari file **logging.txt** dengan referensi [disini](#)

Basic REST API

- ☐ REST API Concept
- ☐ HTTP/HTTPS
- ☐ HTTP Method & HTTP Status Code
- ☐ Response Body
- ☐ Request Payload
- ☐ Endpoint

Basic REST API

- Mengetahui tentang basic REST API
- Mengetahui tentang protokol HTTP dan HTTPS
- Mengetahui tentang HTTP Method dan HTTP Status Code
- Mengetahui tentang penggunaan Response Body
- Mengetahui tentang penggunaan Request Payload
- Mengetahui tentang Endpoint di REST API

Objektif Sesi

- Student memahami tentang basic REST API
- Student mengetahui tentang protokol HTTP dan HTTPS
- Student mengetahui tentang HTTP Method dan HTTP Status Code
- Student mengetahui tentang penggunaan Response Body
- Student mengetahui tentang penggunaan Request Payload
- Student mengetahui tentang Endpoint di REST API

Basic REST API



REST API Concept



HTTP/HTTPS



HTTP Method & HTTP Status Code



Response Body



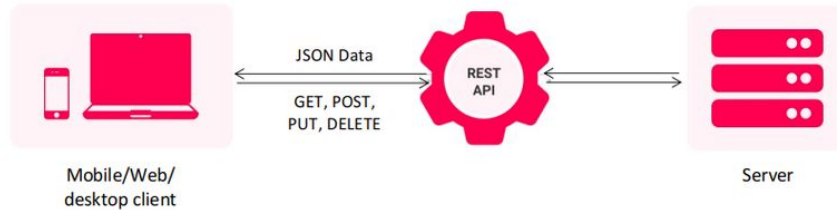
Request Payload



Endpoint

REST API

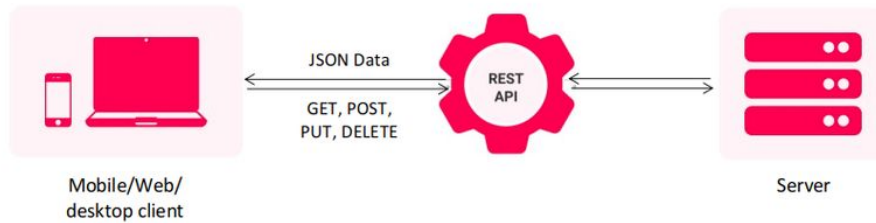
REST API Model



REST API adalah antarmuka yang digunakan oleh dua sistem komputer untuk bertukar informasi melalui internet. Sebagian besar aplikasi bisnis harus berkomunikasi dengan aplikasi internal atau pihak ketiga lainnya untuk melakukan berbagai tugas dan fungsi dari bisnis

API

REST API Model

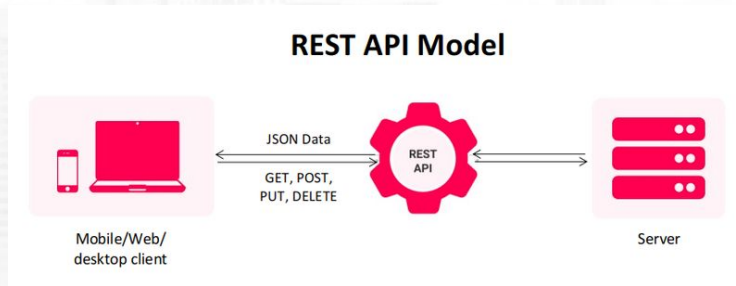


Antarmuka Program Aplikasi (API) menentukan aturan yang harus yang harus diikuti untuk berkomunikasi dengan sistem perangkat lunak lain.

Contohnya, aplikasi absensi mengekspos API yang meminta nama lengkap pegawai dan rentang tanggal.

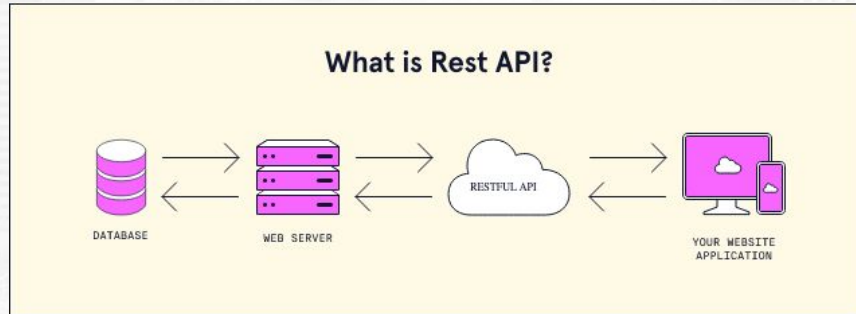
REST

Representational State Transfer (REST) adalah arsitektur perangkat lunak yang memberikan syarat mengenai bagaimana API bekerja.



REST API Working Flow

- Client mengirimkan permintaan ke server
- Server mengautentikasi client
- Server menerima permintaan
- Server mengembalikan respon kepada client



Basic REST API



REST API Concept



HTTP/HTTPS



HTTP Method & HTTP Status Code



Response Body



Request Payload



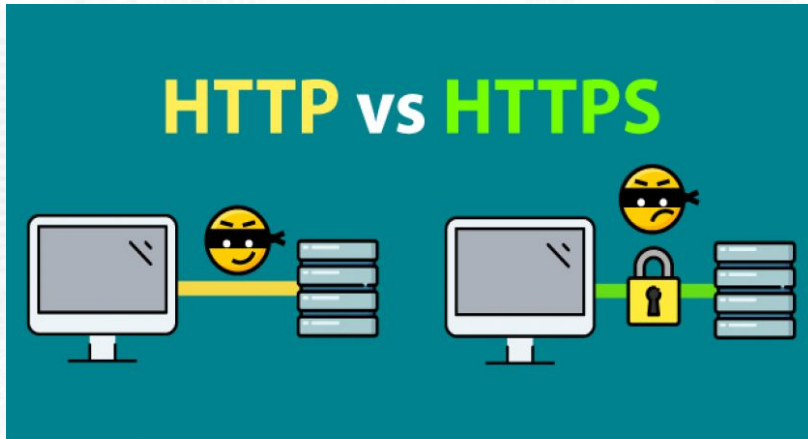
Endpoint

HTTP



HTTP adalah protokol jaringan di level aplikasi atau (application layer) yang dikembangkan untuk membantu proses transfer antar komputer. Protokol ini berguna untuk mentransfer informasi seperti dokumen, file, gambar, dan video antar komputer.

HTTP/S

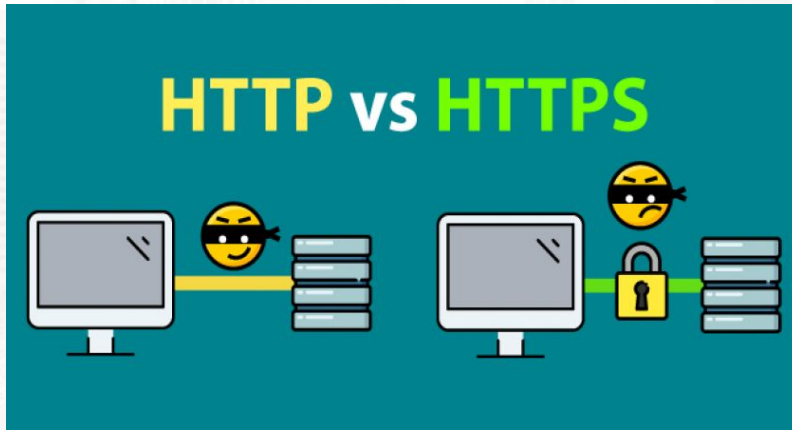


HTTPS adalah singkatan dari Hypertext Transfer Protocol Secure atau versi aman dari Hypertext Transfer Protocol (HTTP).

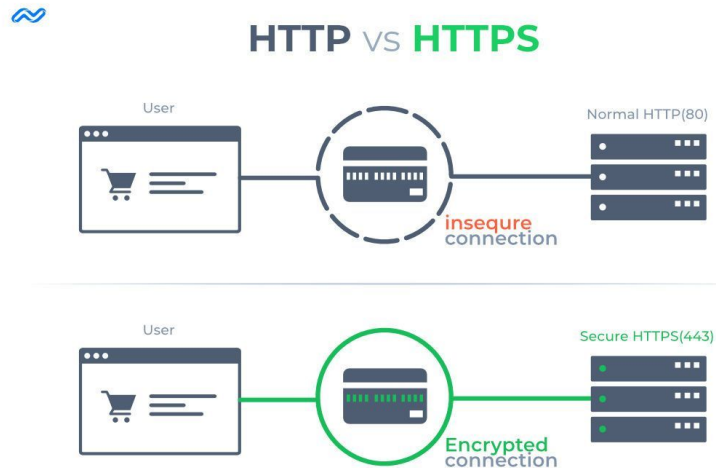
Enkripsi yang dilakukan pada HTTPS adalah pada bagian data session. Proses enkripsi ini menggunakan protokol SSL (Secure Socket layer) atau protokol TLS (Transport Layer Security). Karena menggunakan protokol ini, umumnya port untuk HTTPS adalah 443.

HTTP/S Benefits

- Meningkatkan SEO Website
- Meningkatkan keamanan Website
- Meningkatkan kredibilitas Website



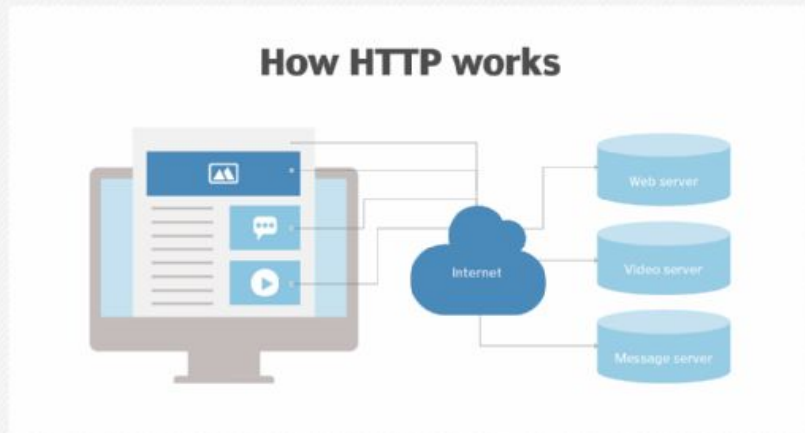
HTTP/S Workflow



- Saat ada permintaan akses ke sebuah website, browser akan mengecek terlebih dahulu apakah website tersebut terhubung dengan sertifikat SSL/TLS atau tidak. (Hal ini tidak terjadi pada koneksi dengan HTTP).
- Jika website tersebut memiliki sertifikat SSL, akan terjadi proses SSL handshake terkait enkripsi dan dekripsi data antara server dan client.
- Nah, kalau sertifikat SSL valid, maka session key akan dibuat oleh server dan client.

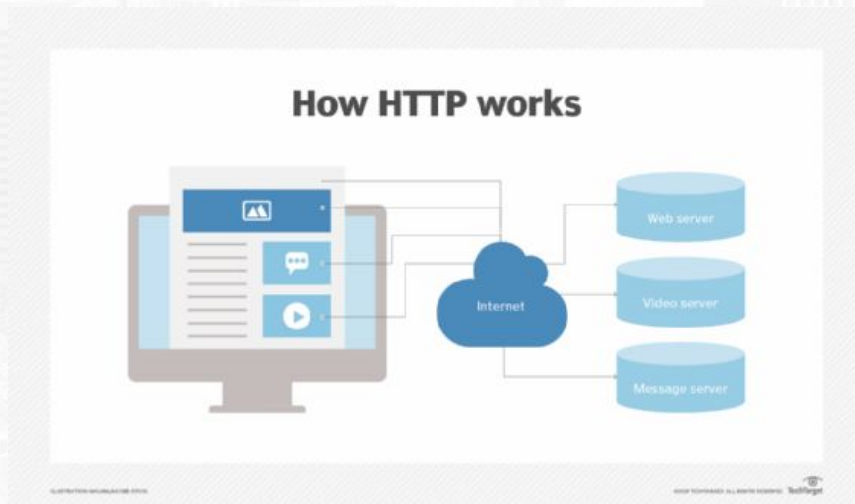
HTTP Benefits

Fungsi HTTP yaitu mengatur format dan bagaimana data ditransmisikan. HTTP juga berfungsi untuk mengatur bagaimana web server dan browser memproses berbagai macam perintah yang masuk.



HTTP Workflow

- HTTP client mengirimkan permintaan informasi ke web server
- HTTP server memproses permintaan client, sedangkan HTTP client menunggu proses selesai
- HTTP server memberikan informasi yang diminta



Basic REST API



REST API Concept



HTTP/HTTPS



HTTP Method & HTTP Status Code



Response Body



Request Payload



Endpoint

HTTP Method



HTTP method adalah mendefinisikan satu set metode permintaan untuk menunjukkan tindakan yang diinginkan untuk dilakukan untuk akses ke dalam server

HTTP Method



- GET - Retrieve data dari server
- POST - Create data di dalam server
- PUT/PATCH - Update data di dalam server
- DELETE - Hapus data di dalam server

HTTP Status Code

HTTP Status Codes



Kode status respon HTTP menunjukkan apakah request dari HTTP telah berhasil atau tidak. Response code dikelompokkan dalam lima bagian yaitu :

2xx - Successful Response

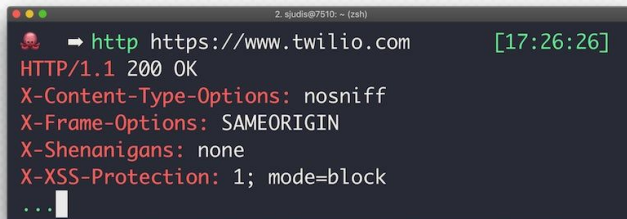
3xx - Redirection response

4xx - Client error response

5xx - Server error response

HTTP Header

HTTP headers for the responsible developer



```

2. sudo@97510: ~ (zsh)
➔ http https://www.twilio.com [17:26:26]
HTTP/1.1 200 OK
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Shenanigans: none
X-XSS-Protection: 1; mode=block
...

```

Header HTTP adalah bidang permintaan atau respons HTTP yang meneruskan konteks dan metadata tambahan tentang permintaan.

Biasanya header berisi informasi yang harus dibawa ketika melakukan request ke dalam server

Basic REST API

- ☒ REST API Concept
- ☒ HTTP/HTTPS
- ☒ HTTP Method & HTTP Status Code



Response Body



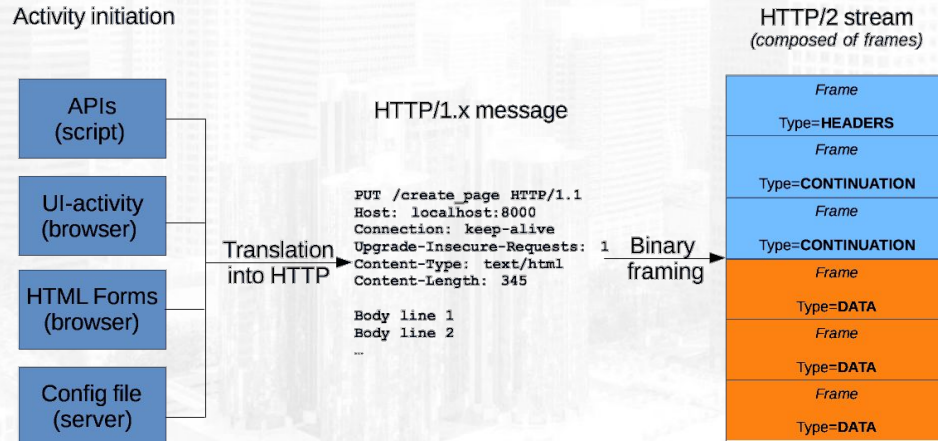
Request Payload



Endpoint

Response Body

Response HTTP adalah bagaimana data dipertukarkan antara server dan client



Response Body

- JSON
- XML
- Text
- Images

```
GET /Myservice/help HTTP/1.1

Host: localhost:8090
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 200 Success
rv0: 3
Content-Type: text/plain; charset=UTF-8
Server: GWS Server (Build 1544519153)
Connection: close
Date: Thu, 10 Jan 2019 17:02:36 GMT
Content-Length: ???
Content-Encoding: ???

Hello world
```

Default header name →

Response body →

Basic REST API

- ☒ REST API Concept
- ☒ HTTP/HTTPS
- ☒ HTTP Method & HTTP Status Code
- ☒ Response Body

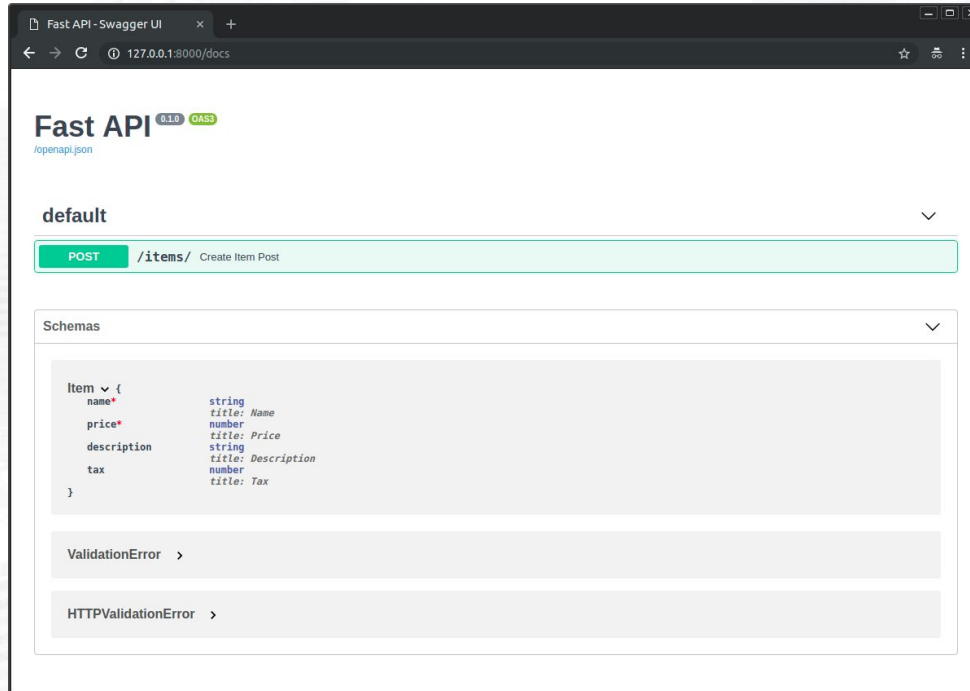


Request Payload



Endpoint

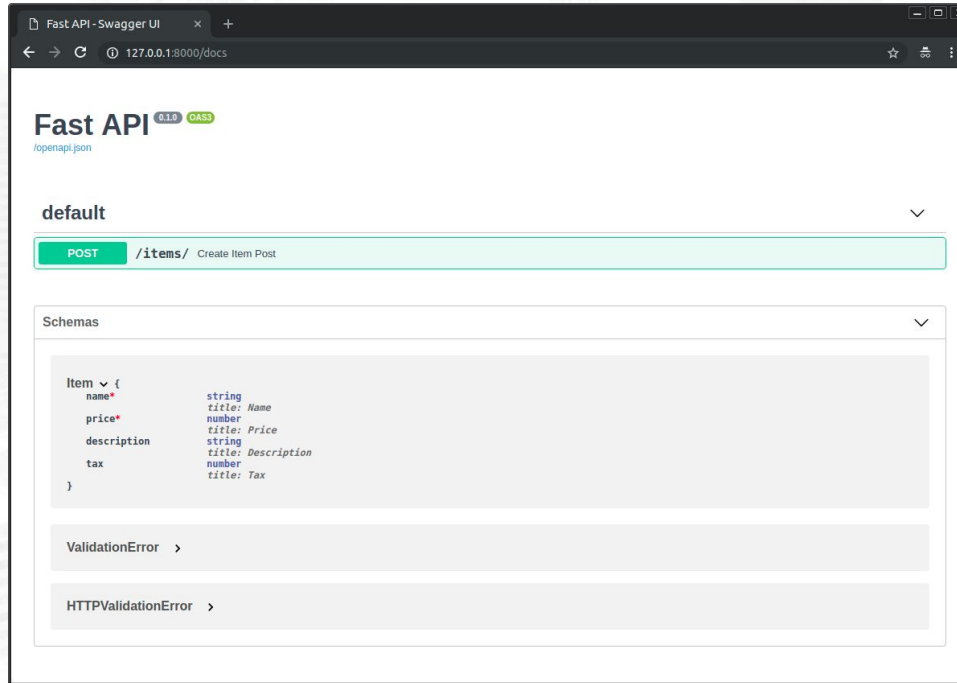
Request Payload



Body Request atau Request Payload adalah bagian dari request HTTP yang dapat dikirim ke server.

Misalnya, jenis file JSON atau XML.

Request Payload



- JSON
- Form Data
- Form-URL-Encoded

Basic REST API

- ☒ REST API Concept
- ☒ HTTP/HTTPS
- ☒ HTTP Method & HTTP Status Code
- ☒ Response Body
- ☒ Request Payload



Endpoint

Endpoint

Changelt

GET `/resources/try` Retrieve Deleteme object created

POST `/resources/try` Does the same as the method below

GET `/resources/try/littletest` a GET Test

default

PUT `/resources/{version}/{context}/entity/{type}`

POST `/resources/{version}/{context}/entity/{type}`

GET `/resources/{version}/{context}/entity/{type}/{id}`

DELETE `/resources/{version}/{context}/entity/{type}/{id}`

Sederhananya, endpoint adalah salah satu jalur komunikasi antara client dan server. Saat API berinteraksi dengan sistem lain, endpoint akan melakukan komunikasi dengan sistem lain

Endpoint Benefits

Changelt

GET	/resources/try	Retrieve Deleteme object created
POST	/resources/try	Does the same as the method below
GET	/resources/try/littletest	a GET Test

default

PUT	/resources/{version}/{context}/entity/{type}
POST	/resources/{version}/{context}/entity/{type}
GET	/resources/{version}/{context}/entity/{type}/{id}
DELETE	/resources/{version}/{context}/entity/{type}/{id}

Di seluruh dunia, perusahaan memanfaatkan API untuk mentransfer informasi penting, proses, transaksi, dan lainnya.

Endpoint Example

Changelt

GET /resources/try Retrieve Deleteme object created

POST /resources/try Does the same as the method below

GET /resources/try/littletest a GET Test

default

PUT /resources/{version}/{context}/entity/{type}

POST /resources/{version}/{context}/entity/{type}

GET /resources/{version}/{context}/entity/{type}/{id}

DELETE /resources/{version}/{context}/entity/{type}/{id}

→ <https://petstore.swagger.io/>

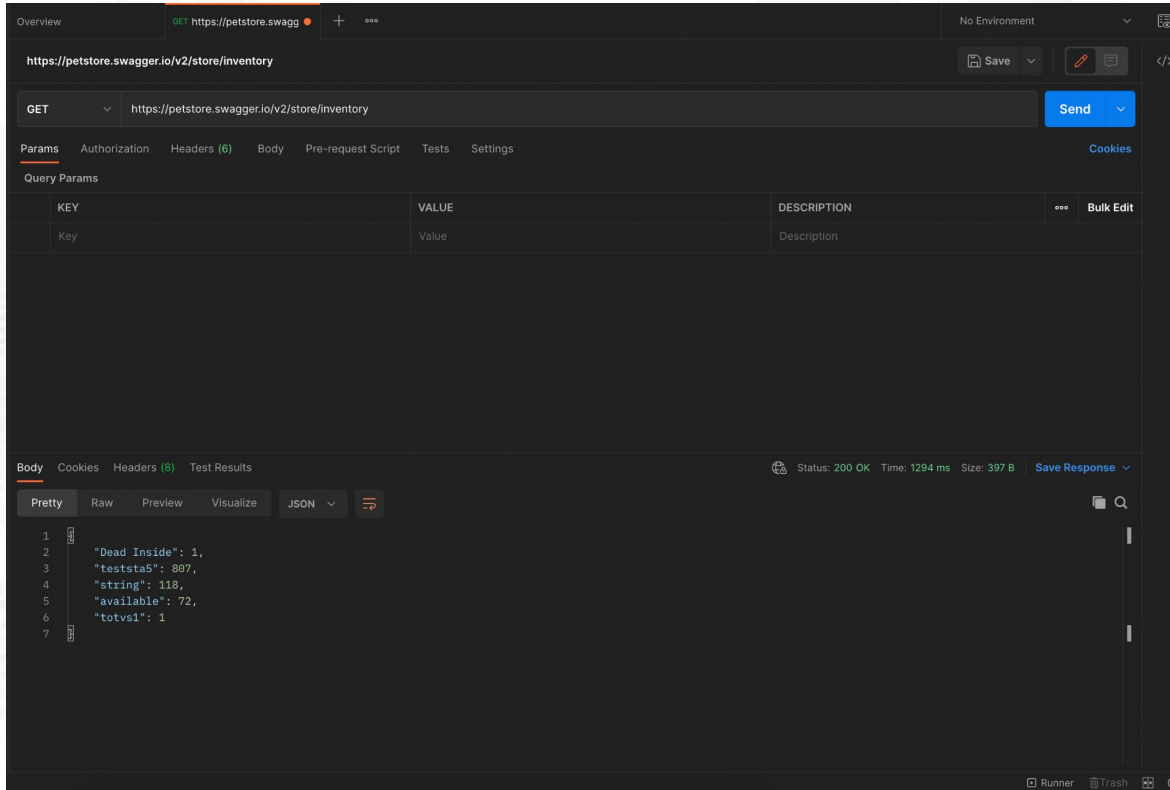
Postman



POSTMAN

Postman adalah platform API bagi pengembang untuk merancang, membangun, menguji dari API yang kita buat.

Endpoint usage on Postman



Overview GET https://petstore.swagger.io/v2/store/inventory No Environment

Save Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
Key	Value	Description	

Body Cookies Headers (6) Test Results

Status: 200 OK Time: 1294 ms Size: 397 B Save Response

Pretty Raw Preview Visualize JSON

```

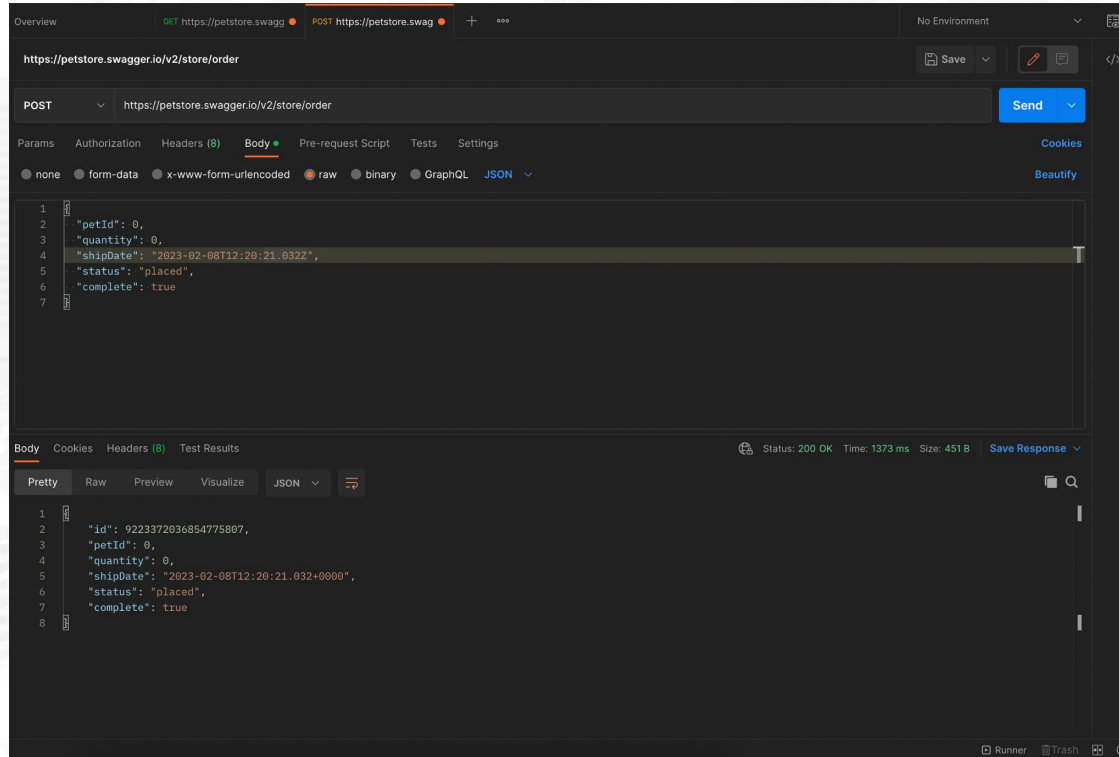
1  {
2    "Dead Inside": 1,
3    "teststa5": 807,
4    "string": 118,
5    "available": 72,
6    "totvs1": 1
7  }

```

Runner Trash

→ Contoh hit
endpoint GET
/store/inventory

Endpoint usage on Postman



→ Contoh hit endpoint
GET
`/store/order`

Study Case

Untuk study case kali ini :

1. Temen temen bisa mulai install Insomnia REST Client atau Postman
2. Setelah selesai instalasi, bisa mulai menggunakan Insomnia atau Postman nya untuk eksplorasi sampling API [disini](#)

Usahakan temen temen mencoba untuk melakukan request ke sample API diatas ya!

Study Case Explained

Untuk penjelasan study case sebelumnya, ada dibawah ini ya

1. Pertama teman teman bisa melakukan instalasi Postman. Untuk proses instalasi nya ada [disini](#) ya
2. Setelah selesai instalasi, teman teman bisa mulai untuk membuka aplikasi Postman nya ya
3. Setelah membuka aplikasi postman nya, teman teman bisa mulai untuk membuka link [berikut](#) untuk sample API
4. Setelah itu teman teman bisa mulai melakukan testing API menggunakan Postman ya

Basic REST API

- ✓ REST API Concept
- ✓ HTTP/HTTPS
- ✓ HTTP Method & HTTP Status Code
- ✓ Response Body
- ✓ Request Payload
- ✓ Endpoint

Reference material

- [REST API Definition](#)
- [HTTPS Definition](#)
- [HTTP Definition](#)
- [HTTP Methods](#)
- [HTTP Status Code](#)
- [HTTP Header](#)
- [HTTP Response](#)
- [Request Body HTTP](#)
- [Endpoint API](#)
- [Fake API](#)

Reference material

- [NodeJS Introduction](#)
- [Runtime Environment Node](#)
- [Runtime Environment Definition](#)
- [Snippet Code Carbon](#)
- [NodeJS Module\(Core, Local and Third party\)](#)
- [Node Package Manager](#)
- [Read and write using Node](#)



Thank you