Js

ติดตั้ง

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, เว็บไซต์

คำอธิบายที่สร้างโดยอัตโนมัติ

<https://nodejs.org/en/>

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, เว็บไซต์, หน้าเว็บ

คำอธิบายที่สร้างโดยอัตโนมัติ

<https://git-scm.com/downloads>

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, ซอฟต์แวร์มัลติมีเดีย

คำอธิบายที่สร้างโดยอัตโนมัติ

ติดตั้ง npm

1 พิมพ์ตามนี้ จะติดตั้งแบบ เซทให้แล้ว

npm init -y

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, ซอฟต์แวร์มัลติมีเดีย

คำอธิบายที่สร้างโดยอัตโนมัติ

2

npm install --save-dev prettier

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, ซอฟต์แวร์มัลติมีเดีย

คำอธิบายที่สร้างโดยอัตโนมัติ

เปลี่ยน

npm run format

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร

คำอธิบายที่สร้างโดยอัตโนมัติ

ลองรัน

const express = require('express');

const app = express();

const port = 3000

app.get('/', (req, res) => { res.send('Hello World!'); });

app.listen(port, () => {

console.log(`Example app listening at http://localhost:${port}`);

});

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, ซอฟต์แวร์มัลติมีเดีย

คำอธิบายที่สร้างโดยอัตโนมัติ

พิมพ์เพื่อติดตั้ง express

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, ซอฟต์แวร์มัลติมีเดีย

คำอธิบายที่สร้างโดยอัตโนมัติ

ผล

ติดตั้ง

npm install express

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, ไลน์

คำอธิบายที่สร้างโดยอัตโนมัติ

node index.js

"start": "node index.js",

ปิดเปิดใหม่หากไม่ได้

npm start

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, จำนวน

คำอธิบายที่สร้างโดยอัตโนมัติ

npm install dotenv

รูปภาพประกอบด้วย ข้อความ, ซอฟต์แวร์, ไอคอนคอมพิวเตอร์, หน้าเว็บ

คำอธิบายที่สร้างโดยอัตโนมัติ

process.env.PORT;

require("dotenv").config();

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ซอฟต์แวร์, แสดง

คำอธิบายที่สร้างโดยอัตโนมัติ

**ทำความรู้จักกับ NodeJS กัน**

**NodeJS** พูดง่าย ๆ กันเลยคือเป็น **JavaScript Open-Source / Cross-Platform Runtime** ในส่วนของ Server / Application ที่ Developer สามารถแก้ไขข้อมูลได้ผ่าน **JavaScript** นั่นเอง ซึ่งเจ้า NodeJS อย่าจำว่าน้องคือ Programming Language หรือ Framework น้องคือสิ่งที่สามารถสร้าง **APIs** ต่าง ๆ ได้ ไม่ว่าจะเป็น Web / Mobile Application จะเห็นได้เลยว่าน้องนั้นเป็นประโยชน์มาก ๆ เลยทีเดียว

**Express.Js เพื่อนรักของ NodeJS**

ในการจัดการกับระบบหลังบ้านสำหรับ Backend Developers นั้น แน่นอนว่าน้อง NodeJS ของเรานั้นเป็นแค่เพียงแค่ตัวเชื่อมเฉย ๆ ไม่ใช่ Language หรือ Framework ดังนั้นเพื่อนที่ชื่อว่า Express.js จะมาช่วยพวกเราเขียนทำให้จัดการกับระบบหลังบ้านได้ง่ายมากขึ้นนั่นเอง โดยที่ Express.js เป็น Framework ตัวหนึ่งของ NodeJS ที่จะช่วยจัดการเรื่องของ APIs ได้อย่างดีเยี่ยม ตัวอย่างการทำงานคร่าว ๆ เช่น

* สามารถเขียน HTTP Methods ได้ใน URL ที่แตกต่างกัน เรียกว่า **Routes**
* สามารถเข้าถึง Object ของ Request , Response และ next function ของ clients ที่ส่งข้อมูลมาหาเราได้ เรียกว่า **Middlewares**
* สามารถใช้ Template Engines หรือ Rendering Data ได้ เช่น static templates file บนหน้าเว็บ HTML ซึ่งง่ายต่อการ Maintenance และ Design หน้าเว็บ ซึ่งใช้ Generator เป็น**view engine**

**ข้อดีหลัก ๆ ของ Express.Js มีดังนี้ :**

* Performance ค่อนข้างดีในการรับ Requests แบบต่อเนื่องหลาย ๆ ครั้ง
* มี Community ของ Express.js ที่เราสามารถนำ APIs , Middleware , Routes ต่าง ๆ มาใช้งานได้เลย ซึ่งจะช่วยประหยัดได้ทั้งต้นทุนและเวลา
* สามารถทำงานกับ Database และ Front-End ได้ ( ขึ้นอยู่กับ Environment ของการทำงาน )
* สามารถสร้าง RESTful APIs ได้ดี ไม่ว่าจะเป็นในเรื่องของ HTTP Requests , Request Validation รวมถึง Response Formatting

**ส่วนข้อเสียของ Express.Js มีดังนี้ :**

* ในบางครั้งอาจไม่ได้ Support ในพวก Specific Projects
* ในเรื่อง Real-Time Application อาจไม่ตอบโจทย์มากเท่าไหร่ ใช้ Socket.IO จะเหมาะสมกว่า

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, แผนภาพ

คำอธิบายที่สร้างโดยอัตโนมัติ