TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN**

**LẬP TRÌNH WEB**

**Tìm hiểu về Node.Js**

*Người hướng dẫn*: **Ths MAI VĂN MẠNH**

*Người thực hiện*: **LÊ THANH BÌNH – 51800015**

**NGUYỄN THỊ HUYỀN TRANG – 51800825**

**TRẦN THU HỒNG – 51800775**

**NGUYỄN QUỐC HUY – 51800780**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN/ĐỒ ÁN CUỐI KÌ MÔN**

**LẬP TRÌNH WEB**

**Tìm hiểu về Node.Js**

*Người hướng dẫn*: **Ths MAI VĂN MẠNH**

*Người thực hiện*: **LÊ THANH BÌNH – 51800015**

**NGUYỄN THỊ HUYỀN TRANG – 51800825**

**TRẦN THU HỒNG – 51800775**

**NGUYỄN QUỐC HUY – 51800780**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2020**

LỜI CẢM ƠN

Báo cáo được hoàn thành tại trường Tôn Đức Thắng. Trong quá trình làm báo cáo chúng em đã nhận được rất nhiều sự giúp đỡ của thầy cô và bạn bè để hoàn thành.

Trước tiên em xin chân thành cám ơn Khoa công nghệ thông tin, trường đại học Tôn Đức Thắng đã tạo điều kiện thuận lợi cho chúng em học tập và thực hiện tiểu luận tìm hiểu về Nodejs.

Em xin bày tỏ lòng biết ơn sâu sắc đến Thầy Đặng Minh Thắng và Mai Văn mạnh đã tận tình truyền đạt kiến thức quý báu cho em trong suốt thời gian học tập vừa qua cũng như đã hướng dẫn cũng như ra những đề tài thú vị để chúng em có thể tiếp cận với bài học một cách tốt hơn.

Mặc dù đã cố gắng hoàn thành đề tài trong phạm vi và khả năng cho phép nhưng không thể tránh khỏi những thiếu sót. Rất mong nhận được sự góp ý và chỉ bảo của quý thầy cô để em rút kinh nghiệm và làm tốt hơn.

Trân trọng cảm ơn!

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của TS Nguyễn Văn A;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 22 tháng 11 năm 2020*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Văn Phước*

*Mai Hữu Bằng*

*Trần Nguyễn Thị Huyền Trang*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm 2020

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm 2020

(kí và ghi họ tên)

TÓM TẮT

Node.js là một nền tảng server-side được xây dựng dựa trên Công cụ JavaScript của Google Chrome (V8 Engine). Mặc dù được phát triển từ Javascript nhưng NodeJS lại có cách làm việc khác. Javascript được nhúng vào HTML, NodeJS hoạt động chủ yếu trên Server dùng để xây dựng các ứng dụng realtime. NodeJS sử dụng mô hình I/O lập trình theo sự kiện non-blocking, do đó node js khá gọn nhẹ và hiệu quả – công cụ hoàn hảo cho các ứng dụng chuyên sâu về dữ liệu theo thời gian thực chạy trên các thiết bị phân tán

Node.js là một môi trường máy chủ mã nguồn mở.

Node.js miễn phí.

Node.js chạy trên nhiều nền tảng khác nhau.

Node.js chạy trên nhiều nền tảng khác nhau (Windows, Linux, Unix, Mac OSX, v.v).

Node.js sử dụng JavaScript trên máy chủ.

MỤC LỤC

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1:Windows Command Prompt 7](#_Toc53247985)

[Hình 2:Câu lệnh trợ giúp Js 7](#_Toc53247986)

[Hình 3:Sau khi cài đặt thành công Node.js 8](#_Toc53247987)

[Hình 4:Kiểm tra phiên bản Node.js 8](#_Toc53247988)

[Hình 5:Ví dụ câu lệnh trong Js 9](#_Toc53247989)

[Hình 7:Câu lệnh chay chương trình 9](#_Toc53247990)

**DANH MỤC CÁC BẢNG BIỂU**

[Bảng 1: Bảng modele tích hợp sẵn trong nodejs 14](#_Toc57686375)

Chương I - Giới thiệu Nodejs

1.1 Khái niệm:

NodeJS Module là một thư viện Javascript, sử dụng kỹ thuật điều khiển theo sự kiện nhập/ xuất không đồng bộ. Nó là một tập hợp các hàm (function) đối tượng và các biến, mà bạn có thể đưa vào ứng dụng của bạn để sử dụng. Sử dụng Module giúp đơn giản việc viết code, và quản lý code trong ứng dụng của bạn. Thông thường mỗi module sẽ được viết trong một tập tin riêng rẽ.

1.2 Nguồn gốc hình thành:

Nodejs được tạo bới Ryan Dahl từ năm 2009, và phát triển dưới sự bảo trợ của Joyent. Mục tiêu ban đầu của Dahl là làm cho trang web có khả năng push như một số ứng dụng web như Gmail. Sau khi thử với vài ngôn ngữ thì Dahl đã chọn JavasScript vì một API nhập/ xuất không đầy đủ. Điều này cho phép có thể định nghĩa một quy ước nhập/ xuất theo sự kiện hay còn gọi là bất đồng bộ (Non-Blocking). Nodejs sử dụng nhiều chi tiết kỹ thuật của CommonJs. Nó cung cấp môi trường REPL cho kiểm thử tương tác.

Có rất nhiều công ty đã sử dụng Node.js để xây dựng cho ứng dụng của họ. Có thể kể một số tên tuổi đình đám như: WalMart, LinkedIn, PayPal, YouTube, Yahoo!, Amazon.com, Netflix, eBay và Reddit.

1.3 Lợi ích mà Nodejs mang lại:

Đầu tiên, V8 JavaScript engine là một Javascript engine mạnh mẽ, được sử dụng trong trình duyệt Chrome của Google. Điều này sẽ làm ứng dụng của bạn có tốc độ rất nhanh.

Node.js khuyến khích viết mã kiểu asynchronous (bất đồng bộ) để cải thiện tốc độ ứng dụng, tránh được những vấn đề phát sinh của việc sử dụng đa luồng.

Nodejs chạy trên nhiều nên tảng khác nhau như Windows, Linux, Unix, Mac OSX, v.v

1.4 Ứng dụng của nodejs và ưu nhược điểm

1.4.1 Ứng dụng của nodejs vào thực tiễn

Node.js có thể tạo nội dung trang động.

Node.js có thể tạo, mở, đọc, ghi, xóa và đóng các tệp trên máy chủ.

Node.js có thể thu thập dữ liệu biểu mẫu.

Node.js có thể thêm, xóa, sửa đổi dữ liệu trong cơ sở dữ liệu của bạn.

1.4.2 Nhược điểm NodeJS

Ứng dụng nặng tốn tài nguyên nếu cần xử lý các ứng dụng tốn tài nguyên CPU như encoding video, convert file, decoding encryption… hoặc các ứng dụng tương tự như vậy thì không nên dùng NodeJS (Lý do: NodeJS được viết bằng C++ & Javascript, nên phải thông qua thêm 1 trình biên dịch của NodeJS sẽ lâu hơn 1 chút). Trường hợp này viết 1 Addon C++ để tích hợp với NodeJS để tăng hiệu suất tối đa!

NodeJS và ngôn ngữ khác NodeJS, PHP, Ruby, Python .NET …thì việc cuối cùng là phát triển các App Web. NodeJS mới sơ khai như các ngôn ngữ lập trình khác.

1.4.3 Ưu điểm NodeJS

Đặc điểm nổi bật của Node.js là nó nhận và xử lý nhiều kết nối chỉ với một single-thread. Điều này giúp hệ thống tốn ít RAM nhất và chạy nhanh nhất khi không phải tạo thread mới cho mỗi truy vấn giống PHP. Ngoài ra, tận dụng ưu điểm non-blocking I/O của Javascript mà Node.js tận dụng tối đa tài nguyên của server mà không tạo ra độ trễ như PHP

JSON APIs Với cơ chế event-driven, non-blocking I/O(Input/Output) và mô hình kết hợp với Javascript là sự lựa chọn tuyệt vời cho các dịch vụ Webs làm bằng JSON.

Ứng dụng trên 1 trang (Single page Application) Ứng dụng thể hiện trên 1 trang gmail - NodeJS rất phù hợp để làm. Với khả năng xử lý nhiều Request/s đồng thời thời gian phản hồi nhanh. NodeJS cho phép các ứng dụng không cần nó tải lại trang, gồm rất nhiều request từ người dùng hoặc cần sự hoạt động nhanh.

NodeJS có thể xử lý hàng nghìn Process và trả ra 1 luồng khiến cho hiệu xuất hoạt động đạt mức tối đa nhất và tuyệt vời nhất.

Streamming Data (Luồng dữ liệu) Các web thông thường gửi HTTP request và nhận phản hồi lại (Luồng dữ liệu). Giả xử sẽ cần xử lý 1 luồng giữ liệu cực lớn, NodeJS sẽ xây dựng các Proxy phân vùng các luồng dữ liệu để đảm bảo tối đa hoạt động cho các luồng dữ liệu khác.

Ứng dụng Web thời gian thực Với sự ra đời của các ứng dụng di động & HTML 5 nên Node.js rất hiệu quả khi xây dựng những ứng dụng thời gian thực (real-time applications) như ứng dụng chat, các dịch vụ mạng xã hội như Facebook, Twitter, …

1.5 Xử lý một yêu cầu tệp:

**Cách node.js xử lý một yêu cầu tệp**

Gửi tác vụ đến hệ thống tệp của máy tính.

Sẵn sàng xử lý các yêu cầu tiếp theo.

Khi hệ thống tệp đã mở và đọc tệp, máy chủ sẽ trả lại nội dung cho máy khách.

Node.js loại bỏ sự chờ đợi và chỉ cần tiếp tục với yêu cầu tiếp theo.

Node.js chạy chương trình đơn luồng, không chặn, không đồng bộ, rất tiết kiệm bộ nhớ.

**Đây là cách PHP hoặc ASP xử lý một yêu cầu tệp.**

Gửi tác vụ đến hệ thống tệp của máy tính.

Chờ trong khi hệ thống tệp mở và đọc tệp.

Trả lại nội dung cho máy khách.

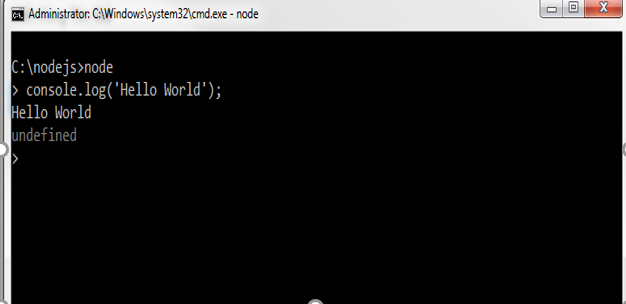
Sẵn sàng xử lý các yêu cầu tiếp theo.

Chương 2 - Cài đặt

2.1 Cài đặt Nodejs

Đầu tiên, bạn truy cập vào liên kết [Node.js](https://nodejs.org/download) để tải về phiên bản Node.js tương thích với hệ điều hành mà bạn đang sử dụng.

Sau khi tải về bạn click đúp vào file cài đặt và tiến hành tiến trình cài đặt. Trong khi cài đặt bạn nên chọn các thông số mặc định sẵn có mà trình cài đặt đưa ra cho bạn. Khi được hỏi xem những tính năng nào bạn muốn cài đặt thì bạn nên chọn cài đặt tất cả trong đó bao gồm cài đặt.

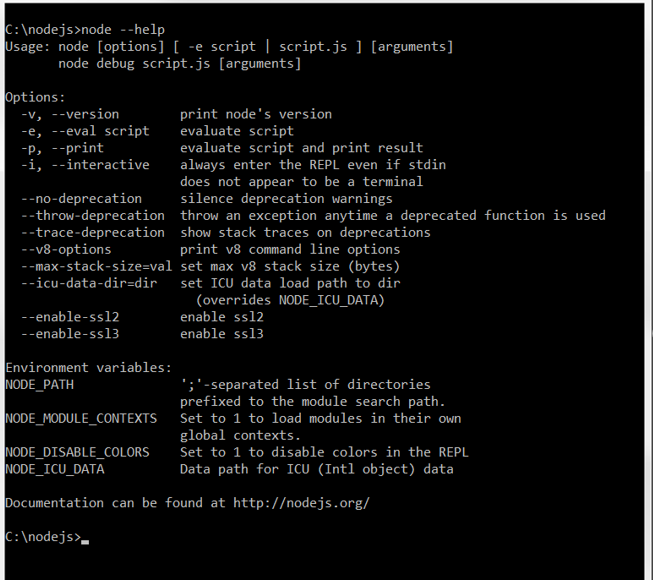


Hình 1 Windows Command Prompt

Sau khi cài đặt xong, bạn mở chương trình **Windows Command Prompt** lên

và chạy câu lệnh: node --help

Nếu theo đúng hướng dẫn trên bạn sẽ thấy màn hình hiển thị kết quả như sau:



Hình 2 Sau khi cài đặt thành công Node.js

Để kiểm tra phiên bản Node.js bạn đang sử dụng, chạy câu lệnh sau:

“node –version”

2.2 Module trong nodejs

****2.2.1 Khái niệm Module trong Nodejs.****

**Module là các đoạn code được đóng gói lại với nhau và được giữ Private, chỉ các hàm và biến được định nghĩa bên trong Module là có thể truy cập và thao tác với nhau. Còn khi nào cần sử dụng Module từ bên ngoài thì chúng ta sẽ chìa các API là các biến, các hàm, hoặc cả 2 biến và hàm ra bên ngoài bằng cách sử dụng đối tượng exports hoặc module.exports.**

Node.js sử dụng kiến trúc Module để đơn giản hóa việc tạo ra các ứng dụng phức tạp. Module là giống như các thư viện trong C, C#, Java, … Mỗi module chứa một tập các hàm chức năng có liên quan đến một “đối tượng” của Module.

Các module được giữ tách biệt riêng với nhau, tách riêng với code base, khi nào cần sử dụng những cái nào thì gọi chúng ra và kết hợp lại với nhau tùy logic xử lý của bạn.

2.2.2 Module tích hợp sẵn trong Nodejs và Module tải về từ bên ngoài

Các module tích hợp (Built-in Module)

**– Các module tích hợp (Built-in Modules):**

Đây là những module mà sau khi cài đặt Nodejs xong bạn đã có thể gọi ra và sử dụng. Ví dụ như http, **https**, **fs**, **path**…vv…

|  |  |
| --- | --- |
| Module | Description |
| assert | Cung cấp tập hợp các assertion tests (Kiểm tra khẳng định) |
| buffer | Để xử lý các dữ liệu nhị phân (binary data). |
| child\_process | Để chạy một tiến trình con (child process) |
| cluster | Để tách một tiến trình (process) thành nhiều tiến trình. |
| crypto | Chứa các hàm mã hóa OpenSSL (OpenSSL cryptographic functions) |
| dgram | Cung cấp các thực hiện (implementation) cho UDP sockets |
| dns | Cung cấp các chức năng để tìm kiếm (lookups) và phân giải (resolution) DNS. |
| events | Để xử lý các sự kiện (events) |
| fs | Để xử lý file system |
| http | Để làm cho Node.js hoạt động như một HTTP server. |
| https | Để làm cho Node.js hoạt động như một HTTPS server. |
| net | Để tạo các server và các client. |
| os | Cung cấp các thông tin về hệ điều hành. |
| path | Để xử lý các đường dẫn file (file paths). |
| querystring | Để xử lý URL query strings |
| readline | Để xử lý các luồng dữ liệu (data streams) mà có thể đọc từng dòng (line) dữ liệu. |
| stream | Xử lý các luồng dữ liệu (streaming data) |
| string\_decoder | Để giải mã (decode) đối tượng bộ đệm (buffer objects) thành string |
| timers | Hẹn giờ để thực thi một hàm Javascript. |
| tls | Để thực hiện các giao thức TLS & SSL. |
| tty | Cung cấp các lớp được sử dụng bởi text terminal. |
| url | Giúp phân tích (parse) các chuỗi URL (URL strings) |
| util | Cung cấp các hàm tiện ích (Utility functions). |
| v8 | Truy cập vào các thông tin của V8 JavaScript engine. |
| vm | Biên dịch (compile) mã JavaScript trong máy ảo (Virtual machine) |
| zlib | Xử lý nén và giải nén các tập tin. |

Bảng 1: Bảng modele tích hợp sẵn trong nodejs

Các module bên ngoài (External Modules)

Node Package Manager - NPM: là trình quản lý gói cho các gói Node.js hoặc các mô-đun nếu bạn muốn. www.npmjs.com lưu trữ hàng ngàn gói miễn phí để tải xuống và sử dụng. Chương trình NPM được cài đặt trên máy tính của bạn khi bạn cài đặt Node.js. Ứng dụng này để cài đặt các module khác. Để cài đặt một module, mở cửa sổ comment line của nodejs ra, vào đường dẫn tương ứng nhập lệnh: npm install module\_name.

Ví dụ viết hoa chữ:



Hình 3 Ví dụ viết hoa chữ

**Tải lên Tệp:**

**B1: Tạo biểu mẫu tải lên**



Hình 4 Tạo biểu mẫu tải lên

**B2: Phân tích cú pháp tệp đã tải lên**



Hình 5 Phân tích cú pháp tệp đã tải lên

**B3: Lưu tệp**



Hình 6 Lưu tệp

2.2.3 Modules Buil-in trong Nodejs: HTTP – URL – File System

2.2.3.1 HTTP module

**http** có thể hiểu đơn giản là module cho phép Nodejs truyền dữ liệu qua giao thức http, cụ thể hơn xíu, chức năng của nó là tạo một máy chủ HTTP lắng nghe các cổng kết nối trên đó và trả lời lại cho máy khách (client).

Phương thức tạo server của http:

**createServer()**: tạo một máy chủ HTTP

const http = require**(**'http'**)**;

const server = http.createServer**((**req, res**)** => **{**

res.write**(**'<h1> Hi, thank you for visit https://abc.com </h1><hr>'**)**;

res.end**()**;

**})**;

server.listen**(**8017, 'localhost'**)**;

Trong method **createServer()** mình có truyền vào 2 tham số:

– **req**: viết tắt của **request**, là biến chứa các thông tin mà Client truyền lên Server.

– **res**: viết tắt của **response**, là biến chứa các thông tin mà Server trả về cho Client

– Một số phương thức bên trong **request (req)** và **response (res)**:

**res.writeHead():**

Sử dụng phương thức này để định nghĩa kiểu dữ liệu trả về. Chúng ta vẫn sử dụng code ví dụ như ở trên, thêm 1 dòng **writeHead()**vào để định nghĩa kiểu dữ liệu trả về là **html**.

const http = require**(**'http'**)**;

const server = http.createServer**((**req, res**)** => **{**

res.writeHead**(**200, **{**'Content-Type': 'text/html'**})**;

res.write**(**'<h1> Hi, thank you for visit https://abc.com </h1><hr>'**)**;

res.end**()**;

**})**;

server.listen**(**8017, 'localhost'**)**;

**res.write()**

Thiết lập nội dung mà server muốn trả về cho client, có thể thêm nhiều dòng **res.write()** liên tiếp nhau được. Ví dụ:

const http = require**(**'http'**)**;

const server = http.createServer**((**req, res**)** => **{**

res.writeHead**(**200, **{**'Content-Type': 'text/html'**})**;

res.write**(**'<html>'**)**;

res.write**(**'<head>'**)**;

res.write**(**'<title>Test module HTTP - Hello</title>'**)**;

res.write**(**'</head>'**)**;

res.write**(**'<body>'**)**;

res.write**(**'<h1> Hi, thank you for visit https://abc.com </h1><hr>'**)**;

res.write**(**'</body>'**)**;

res.write**(**'</html>'**)**;

res.end**()**;

**})**;

server.listen**(**8017, 'localhost'**)**;

**req.url()**

Bên trong đối tượng **request** (req) từ client truyền lên có một thuộc tính là **url**, thuộc tính này chứa các phần tử phía sau tên miền của bạn, ví dụ:

const http = require**(**'http'**)**;

const server = http.createServer**((**req, res**)** => **{**

res.writeHead**(**200, **{**'Content-Type': 'text/html'**})**;

res.write**(**req.url**)**;

res.end**()**;

**})**;

server.listen**(**8017, 'localhost'**)**;

2.2.3.2 URL module

Chức năng của module này là **chia nhỏ địa chỉ trang web** thành các phần có thể đọc được, phục vụ cho nhiều tác vụ cần lấy dữ liệu từ url.

Giả sử mình truyền lên server một url như thế này:

**http://localhost:8017/search.html?animal=cat&color=black**

Bây giờ từ server muốn lấy ra 2 giá trị là **cat** và **black**, mình sẽ code như sau:

const http = require**(**'http'**)**;

const url = require**(**'url'**)**;

const server = http.createServer**((**req, res**)** => **{**

res.writeHead**(**200, **{**'Content-Type': 'text/html'**})**;

let urlData = url.parse**(**req.url, **true)**;

console.log**(**urlData**)**;

res.write**(**`param animal: ${ urlData.query.animal } <br>`**)**;

res.write**(**`param color: ${ urlData.query.color } <br>`**)**;

res.end**()**;

**})**;

server.listen**(**8017, 'localhost'**)**;

2.2.3.3 File System module

**fs** là viết tắt của file system, module này cho phép chúng ta làm việc với các file ở trên máy tính của mình.

Những trường hợp mà hàng ngày ta hay làm như:

Tạo file – Đọc file – Cập nhật file – Đổi tên file – Xóa file.

**Tạo file:  fs.writeFile();**

Trong project mình tạo thêm một thư mục là **files** để lưu trữ các file mà chuẩn bị mình sẽ tạo bằng code.

– Đầu tiên là tạo một file **cat.txt** đơn giản, lưu tên vài con mèo chẳng hạn, code sẽ trông như sau:

const fs = require**(**'fs'**)**;

let fileContent = 'Ragdoll, Scottish fold, British shorthair...';

let filePath = 'files/cats.txt';

fs.writeFile**(**filePath, fileContent, **(**err**)** => **{**

**if(**err**)** **throw** err;

console.log**(**'The file was successfully saved.'**)**;

**})**;

Trong đoạn code trên, ta nạp module **fs**, tạo 2 biến **fileContent** và **filePath** để định nghĩa nội dung và đường dẫn của file cần tạo.

Cuối cùng là dùng phương thức **fs.writeFile()**với 3 tham số truyền vào lần lượt là **filePath**, **fileContent** và một **callback function.**

Phương thức này sẽ ghi dữ liệu vào một tệp, nếu như tệp đó chưa tồn tại thì sẽ tự động tạo mới, còn nếu đã có thì sẽ ghi đè dữ liệu lên. Dữ liệu có thể là chuỗi ký tự (string) hoặc kiểu buffer.

Kết quả sau khi chạy file server.js: node server.js, tệp **cat.txt** được tạo thành công.

**Đọc file: fs.readFile()**

Bây giờ, trong thư mục **files**của dự án, tạo thêm 1 file là **cat.html** như sau

<!DOCTYPE html>

**<html** lang="en"**>**

**<head>**

**<meta** charset="UTF-8"**>**

**<meta** name="viewport" content="width=device-width, initial-scale=1.0"**>**

**<meta** http-equiv="X-UA-Compatible" content="ie=edge"**>**

**<title>**Cats**</title>**

**</head>**

**<body>**

This is sample content. Thank you for reading!

**<p>**

**<span** style="font-weight:bold;"**>**Hello:**</span>**

**<a** href="https://abc.com/"target="\_blank"**>**https://abc.com/**</a>**

**</p>**

**</body>**

**</html>**

Và ở file **server.js**, mình sửa lại code để đọc file **cat.html** sau đó trả dữ liệu về trình duyệt:

const http = require**(**'http'**)**;

const fs = require**(**'fs'**)**;

const server = http.createServer**((**req, res**)** => **{**

let filePath = 'files/cat.html';

fs.readFile**(**filePath, **(**err, data**)** => **{**

**if(**err**)** **throw** err;

res.writeHead**(**200, **{**'Content-Type': 'text/html'**})**;

res.write**(**data**)**;

res.end**()**;

**})**;

**})**;

server.listen**(**8017, 'localhost'**)**;

**Cập nhật file: fs.appendFile()**

Như ở phần tạo file bằng phương thức **fs.witeFile()** ở trên, chúng ta có thể dùng chính nó để làm mới lại toàn bộ nội dung của một file.

Một phương thức nữa là **fs.appendFile()**, phương thức này sử dụng khi chúng ta muốn thêm dữ liệu vào cuối file.

Ví dụ muốn thêm một con mèo nữa là “Korat cat” vào file **cats.txt**

const http = require**(**'http'**)**;

const fs = require**(**'fs'**)**;

let content = 'Korat cat';

let filePath = 'files/cat.txt';

fs.appendFile**(**filePath, content, **(**err**)** => **{**

**if(**err**)** **throw** err;

console.log**(**'The file was successfully updated.'**)**;

**})**;

**Đổi tên file: fs.rename()**

Giờ mình lại muốn đổi cái tên **“cats.txt”** thành **“những con mèo.txt”**

const http = require**(**'http'**)**;

const fs = require**(**'fs'**)**;

fs.rename**(**'files/cats.txt', 'files/những con mèo.txt', **(**err**)** => **{**

**if(**err**)** **throw** err;

console.log**(**'The file was successfully renamed.'**)**;

**})**;

**Xóa file: fs.unlink()**

const http = require('http');

const fs = require('fs');

fs.unlink('files/những con mèo.txt', (err) => {

if(err) throw err;

console.log('The file was successfully deleted.');

});

2.3 Tạo và chạy chương trình Node.js

Giống như các chương trình viết bằng những ngôn ngữ lập trình khác thì chương trình viết bằng Node.js cũng được lưu dưới dạng tệp tin văn bản đơn giản được tạo ra sử dụng các chương trình viết mã lệnh như Nodepad hay Nodepad++ (chú ý: chúng ta không sử dụng chương trình Microsoft Office như Word để viết chương trình Node.js).

**Bước 1**: Mở chương trình viết mã lệnh. Trên Windows bạn có thể sử dụng Nodepad (hoặc Nodepad++ nếu như bạn đã cài đặt nó) để viết mã lệnh cho các chương trình Node.js

**Bước 2**: Tạo tệp tin chương trình. Trên Nodepad (hoặc Nodepad++) bạn tạo tệp tin mới bằng cách sử dụng bấm tổ hợp phím Ctrl + N (hoặc ở danh sách menu chọn File => New file)

**Bước 3**: Viết mã lệnh. Trong tệp tin mới tạo ra bạn nhập vào nội dung dưới đây:

Bấm tổ hợp phím Ctrl + S (hoặc ở danh sách menu chọn File => Save) và trong cửa số bật lên bạn nhập tên tập tin là hello.js sau đó lưu lại.

Console.log(“Xin chào”);

**Bước 4**: Chạy chương trình. Để chạy chương trình vừa mới tạo ở trên bạn hãy mở command prompt lên và chuyển tới thư mục chứa tệp hello.js ở trên. Để di chuyển giữa các thư mục trên command prompt, bạn sử dụng lệnh cd (viết tắt của cụm từ change directory). Ví dụ nếu bạn lưu tệp tin trên ở thư mục C:\nodejs\hello.js bạn có thể di chuyển tới thư mục này bằng cách nhập vào câu lệnh sau trên cửa sổ command prompt: cd C: \nodejs\ .

Sau đó bấm Enter để di chuyển tới thư mục này. Và cuối cùng để chạy chương trình trong tệp hello.js, bạn nhập vào và chạy câu lệnh sau: node hello.js

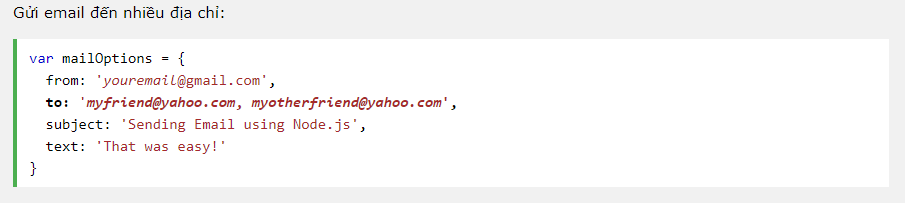
2.4 Gửi email:

Bây giờ bạn đã sẵn sàng để gửi email từ máy chủ của mình.

Sử dụng tên người dùng và mật khẩu từ nhà cung cấp email đã chọn của bạn để gửi email. Hướng dẫn này sẽ chỉ cho bạn cách sử dụng tài khoản Gmail của bạn để gửi email:



Nhiều người nhận:



Gửi HTML

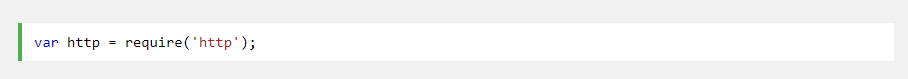
Để gửi văn bản có định dạng HTML trong email của bạn, hãy sử dụng thuộc tính "html" thay vì thuộc tính "văn bản":



4

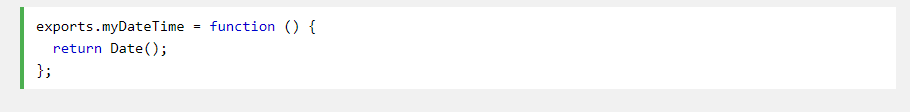
6. Node.js Modules

**Để dùng ta gõ câu lệnh như sau:**



**Bây giờ ứng dụng đã có quyền truy cập vào mô đun http:**

**Tạo mô-đun riêng:**



Sử dụng exportstừ khóa để cung cấp các thuộc tính và phương thức bên ngoài tệp mô-đun. Lưu mã ở trên trong một tệp có tên "myfirstmodule.js"



**Thêm tiêu đề HTTP**



Đối số đầu tiên của res.writeHead() phương thức là mã trạng thái, 200 nghĩa là tất cả đều OK, đối số thứ hai là một đối tượng chứa các tiêu đề phản hồi.

**Đọc chuỗi truy vấn**

Hàm được truyền vào http.createServer() có một reqđối số đại diện cho yêu cầu từ máy khách, là một đối tượng (đối tượng http.IncomingMessage).

Đối tượng này có một thuộc tính gọi là "url" chứa một phần của url đứng sau tên miền:



**Tách chuỗi truy vấn**

Có các mô-đun tích hợp sẵn để dễ dàng chia chuỗi truy vấn thành các phần có thể đọc được, chẳng hạn như mô-đun URL.



TÀI LIỆU THAM KHẢO

Tiếng Việt

Tiếng Anh