Introdução ao Node.JS

Usabilidade, desenvolvimento web, mobile e jogos

Prof. Me. Gustavo Torres Custódio
gustavo.custodio@ulife.com.br

Conteúdo

HTTP

Formulários

Exercícios

Node.JS

- Node.JS é um ambiente de código aberto para criação de web servers.

Node.JS

- Node.JS é um ambiente de código aberto para criação de web servers.
  - Construído em JavaScript.
  - Ele é uma alternativa para tecnologias como PHP ou Java.

Node.JS

- Multiplataforma: permite criar desde aplicativos desktop, aplicativos móveis etc...

Node.JS

- Multiplataforma: permite criar desde aplicativos desktop, aplicativos móveis etc...
- Multi-paradigma: é possível programar em diferentes paradigmas, como: Orientado a Objetos, funcional, imperativo e dirigido à eventos;

Node.JS

- Open Source: é uma plataforma de código aberto, isso significa que você pode ter acesso ao código fonte do Node.JS e realizar suas próprias customizações ou mesmo contribuir para a comunidade de forma direta;

Node.JS

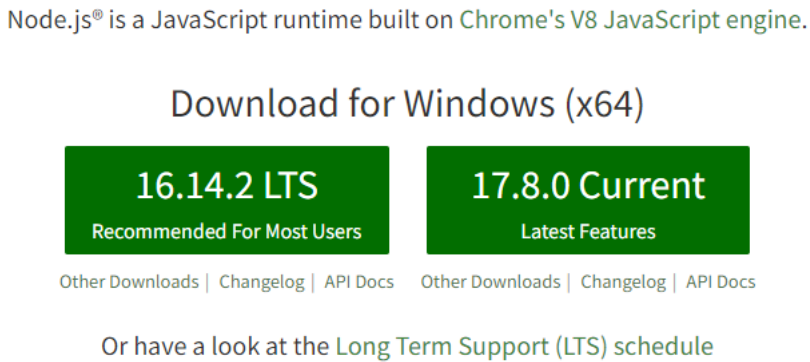
- Open Source: é uma plataforma de código aberto, isso significa que você pode ter acesso ao código fonte do Node.JS e realizar suas próprias customizações ou mesmo contribuir para a comunidade de forma direta;
- Escalável: Node.JS foi criado para construir aplicações web escaláveis, como podemos ver na sua documentação oficial.

Instalação

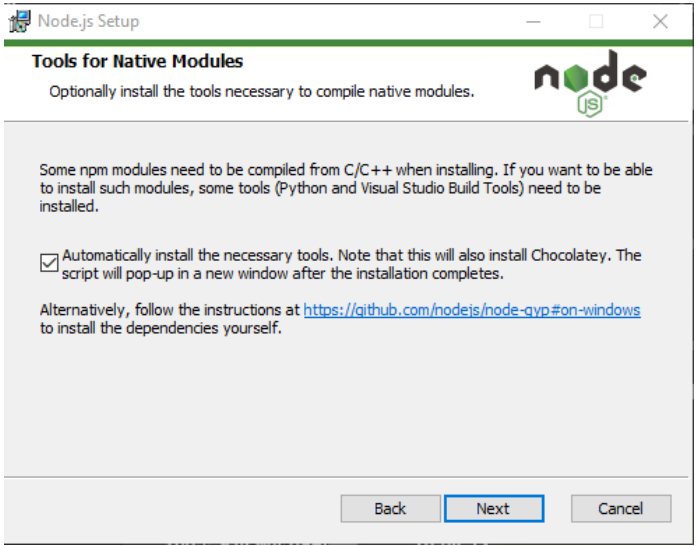
- https://nodejs.org/

Instalação

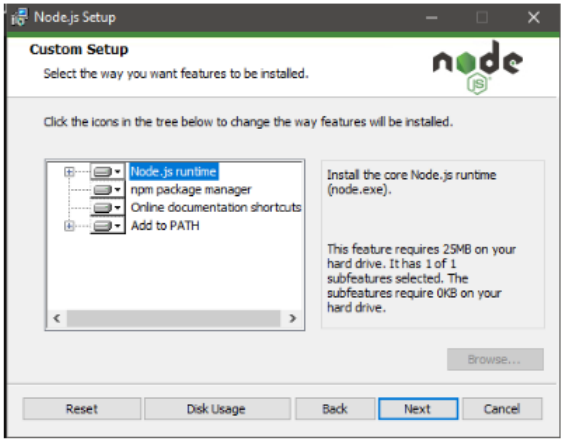
- https://nodejs.org/



Instalação


Node.js Setup
Tools for Native Modules
Optionally install the tools necessary to compile native modules.
Some npm modules need to be compiled from C/C++ when installing. If you want to be able to install such modules, some tools (Python and Visual Studio Build Tools) need to be installed.
Automatically install the necessary tools. Note that this will also install Chocolatey. The script will pop-up in a new window after the installation completes.
Alternatively, follow the instructions at https://github.com/nodejs/node-gyp#on-windows to install the dependencies yourself.
Back
Next
Cancel

Instalação


Node.js Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
Node.js runtime
npm package manager
Online documentation shortcuts
Add to PATH
Install the core Node.js runtime (node.exe).
This feature requires 25MB on your hard drive. It has 1 of 1 subfeatures selected. The subfeatures require 0KB on your hard drive.
Browse...
Reset
Disk Usage
Back
Next
Cancel

Primeiro Servidor

- Vamos criar um primeiro servidor utilizando Node.JS:

Primeiro Servidor

- Vamos criar um primeiro servidor utilizando Node.JS:
  - Abra seu editor de texto favorito e crie o arquivo **app.js**:

Primeiro Servidor

- Vamos criar um primeiro servidor utilizando Node.JS:
  - Abra seu editor de texto favorito e crie o arquivo **app.js**:

```
const http = require('http');
const hostname = '127.0.0.1'; // localhost
const port = 3333; // Porta da aplicação

const server = http.createServer((req, res) => {
    res.statusCode = 200; // status retornado indicando sucesso
    res.setHeader('Content-Type', 'text/html');
    res.end('<h1>Hello World</h1>');
});

server.listen(port, hostname, () => {
    console.log('Server running at http://' + hostname + ':' + port + '/');
});
```

Primeiro Servidor

- Execute no terminal com a instrução ‘node app.js’.
  - Não esqueça de estar no mesmo diretório do arquivo app.js.

Primeiro Servidor

- Execute no terminal com a instrução ‘node `app.js`’.
  - Não esqueça de estar no mesmo diretório do arquivo `app.js`.
- O servidor recebe uma solicitação do navegador na porta 3333..

Primeiro Servidor

- Execute no terminal com a instrução ‘node `app.js`’.
  - Não esqueça de estar no mesmo diretório do arquivo `app.js`.
- O servidor recebe uma solicitação do navegador na porta 3333..
  - Se tudo der certo, o servidor devolve uma mensagem com **status 200**.
    - Código para Ok.

Primeiro Servidor

- Execute no terminal com a instrução ‘node `app.js`’.
  - Não esqueça de estar no mesmo diretório do arquivo `app.js`.
- O servidor recebe uma solicitação do navegador na porta 3333..
  - Se tudo der certo, o servidor devolve uma mensagem com **status 200**.
    - Código para Ok.
  - A mensagem retornada contém uma tag <**h1**> com a frase “Hello World”.

Introducao ao Node.JS

HTTP

GET e POST

- O HTTP é **o protocolo responsável pela comunicação na web**.

GET e POST

- O HTTP é **o protocolo responsável pela comunicação na web**.
  - Por padrão, quando acessamos um site, esse é o protocolo utilizado.
  - Esse protocolo possui um conjunto de métodos de acesso (verbos).
  - Os dois verbos mais comuns do HTTP são o **POST e o GET**.

GET e POST

- O GET é o verbo padrão utilizado quando requisitamos qualquer informação para um servidor.

GET e POST

- O GET é o verbo padrão utilizado quando requisitamos qualquer informação para um servidor.
  - Ele envia todas as informações para o servidor na URL.

GET e POST

- O GET é o verbo padrão utilizado quando requisitamos qualquer informação para um servidor.
  - Ele envia todas as informações para o servidor na URL.
  - Essa informação é facilmente visível.
    - Arriscado dependendo da situação.

GET e POST

- O POST envia os dados para o servidor dentro do corpo da requisição HTTP.

GET e POST

- O POST envia os dados para o servidor dentro do corpo da requisição HTTP.
  - Torna os dados mais difíceis de serem visualizados.
  - No entanto, ainda são visíveis.

GET e POST

- O POST envia os dados para o servidor dentro do corpo da requisição HTTP.
  - Torna os dados mais difíceis de serem visualizados.
  - No entanto, ainda são visíveis.
  - A forma correta de esconder os dados do formulário é por meio do protocolo HTTPS.
    - Por isso sites em HTTP informam que a conexão não é segura.

Introducao ao Node.JS

Formulários

Formulários e POST

- Em formulários, é comum a presença de informações sensíveis.

Formulários e POST

- Em formulários, é comum a presença de informações sensíveis.
  - Por isso o POST tem preferência como verbo nesse caso.
    - `<form action="destino" method="POST">`

Formulários e POST

- Em formulários, é comum a presença de informações sensíveis.
  - Por isso o POST tem preferência como verbo nesse caso.
    - `<form action="destino" method="POST">`
- Vamos construir um servidor web em Node que recebe solicitações GET:

Formulários e POST

- Em formulários, é comum a presença de informações sensíveis.
  - Por isso o POST tem preferência como verbo nesse caso.
    - `<form action="destino" method="POST">`
- Vamos construir um servidor web em Node que recebe solicitações GET:
  - Crie um app chamado getrequest.js.
  - Execute ele com "node `getrequest.js`".

Formulários e POST

- Em formulários, é comum a presença de informações sensíveis.
  - Por isso o POST tem preferência como verbo nesse caso.
    - `<form action="destino" method="POST">`
- Vamos construir um servidor web em Node que recebe solicitações GET:
  - Crie um app chamado getrequest.js.
  - Execute ele com "`node getrequest.js`".
  - Instale a biblioteca express:
    - `npm install express`.

Express

- Express é um framework para Node.JS que permite **criar aplicativos da web de forma mais fácil e rápida**.
  - É amplamente utilizado na comunidade Node.js para desenvolvimento de aplicativos da web.

Formulários e POST

```
var express = require('express');
var http = require('http');

var app = express();
var server = http.createServer(app);

// Se digitarmos no navegador localhost:3333, faremos uma requisição GET
app.get('/', function(req, res){
    // Servidor manda a resposta da requisição de volta para o cliente
    res.send("<h1>Você se conectou com sucesso</h1>");
});

// Esperando na porta 3333
server.listen(3333, function(){
    console.log("Server listening on port: 3333");
});
```

Formulários e POST

- Observe o primeiro parâmetro do get (/).

Formulários e POST

- Observe o primeiro parâmetro do get (/).
  - Isso significa que estamos na raiz do servidor.
- Suponha que desejamos mudar a URL para:

Formulários e POST

- Observe o primeiro parâmetro do get (/).
  - Isso significa que estamos na raiz do servidor.
- Suponha que desejamos mudar a URL para:
  - `localhost:3333/metodoget`.
  - Para isso, apenas mudamos um trecho do código:
    - `app.get('/metodoget'...`

Formulários e POST

- Com a mesma facilidade é possível alterar a função para receber métodos POST.

Formulários e POST

- Com a mesma facilidade é possível alterar a função para receber métodos POST.
  - `app.post('/metodopost', function(req, res)\{.`

Formulários e POST

- Com a mesma facilidade é possível alterar a função para receber métodos POST.
  - `app.post('/metodopost', function(req, res)\{.`
- Observe que para requisições POST, não conseguimos acessar o servidor pela URL no navegador.
  - Isso porque quando entramos em um site pelo navegador, realizamos uma **requisição GET.**

Formulários e POST

- Uma forma de realizar uma requisição POST é por meio do envio de um formulário HTML (com `method POST`).

Formulários e POST

- Uma forma de realizar uma requisição POST é por meio do envio de um formulário HTML (com `method POST`).
- Enviaremos um `<form>` HTML para o servidor.

Formulários e POST

- Vamos criar um formulário para inserir informações de um funcionário.

Formulários e POST

- Vamos criar um formulário para inserir informações de um funcionário.
- Crie um arquivo **`formulario.html`**.

Formulários e POST

- Vamos criar um formulário para inserir informações de um funcionário.
- Crie um arquivo **`formulario.html`**.

```
<form action="/mostrar" method="POST">
    <fieldset id="dados">
        <legend>Dados</legend>
        <p> <label for="id">Id:</label>
            <input type="number" name="id" id="id"> </p>

        <p> <label for="nome">Nome:</label>
            <input type="text" name="nome" id="nome"> </p>

        <p> <label for="idade">Idade:</label>
            <input type="number" name="idade" id="idade"> </p>

        <p> <label for="endereco">Endereço:</label>
            <input type="text" name="endereco" id="endereco"> </p>
    </fieldset>
    <button type="submit">OK</button>
</form>
```

Formulários e POST

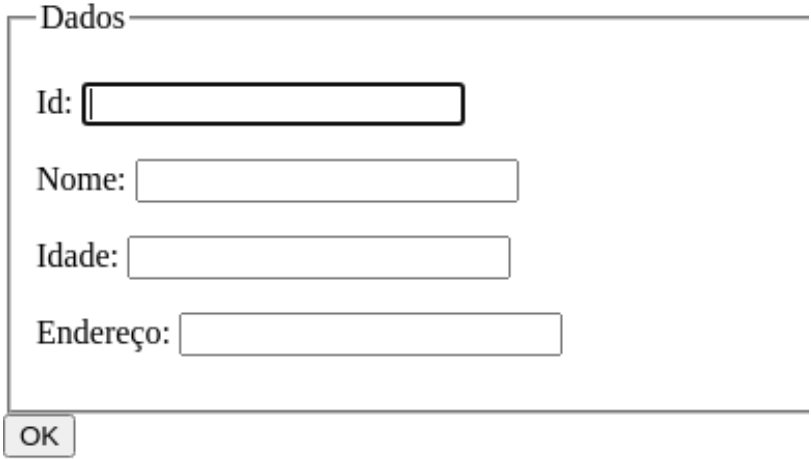
- Precisamos executar o esse formulário HTML dentro do nosso servidor:
  - Para isso criamos um arquivo `formulario.js` e criamos o método para atender requests GET:

Formulários e POST

- Precisamos executar o esse formulário HTML dentro do nosso servidor:
  - Para isso criamos um arquivo `formulario.js` e criamos o método para atender requests GET:

```
var path = require("path");
app.get('/', function(req,res){
    /* Envia como resposta o formulário html.
     * __dirname éa pasta do arquivo atual.
     */
    res.sendFile(path.join(__dirname,'./formulario.html'));
});
```

Formulários e POST



Formulários e POST

- Quando o formulário é enviado, é necessário ter o servidor esperando por uma requisição do tipo POST.

Formulários e POST

- Quando o formulário é enviado, é necessário ter o servidor esperando por uma requisição do tipo POST.
- O servidor vai processar uma requisição do tipo POST e enviar uma página de resposta para o usuário.

Formulários e POST

- Quando o formulário é enviado, é necessário ter o servidor esperando por uma requisição do tipo POST.
- O servidor vai processar uma requisição do tipo POST e enviar uma página de resposta para o usuário.
- A resposta é uma página HTML.

Formulários e POST

- Agora faremos um exemplo em que enviamos um formulário e o servidor devolve as informações contidas nele em parágrafos.

Formulários e POST

- Agora faremos um exemplo em que enviamos um formulário e o servidor devolve as informações contidas nele em parágrafos.
- Instale o **body parser**:
  - `npm install body-parser`.
  - Ele é utilizado para processar informações dentro do corpo de um formulário.

Formulários e POST

- Adicione todas as importações necessárias:

Formulários e POST

- Adicione todas as importações necessárias:

```
var express = require('express');
var http = require('http');
var bodyParser = require('body-parser');
var path = require("path");
var app = express();
```

Formulários e POST

```
app.use(bodyParser.urlencoded({extended: false}));

var server = http.createServer(app);
// Se digitarmos no navegador localhost:3333, faremos uma requisição POST
app.post('/mostrar', function(req, res){
    res.send("<p>Nome: " + req.body.nome + "</p>" +
             "<p>Idade: " + req.body.idade + "</p>" +
             "<p>Endereço: " + req.body.endereco + "</p>"
            );
});
server.listen(3333, function(){
    console.log("Server listening on port: 3333");
});
```

Formulários e POST

- As informações passadas em um formulário ficam contidas dentro do corpo da requisição:

Formulários e POST

- As informações passadas em um formulário ficam contidas dentro do corpo da requisição:
  - `req.body`.

Formulários e POST

- As informações passadas em um formulário ficam contidas dentro do corpo da requisição:
  - `req.body`.
- Se queremos recuperar informações do atributo **nome**, utilizamos a instrução:

Formulários e POST

- As informações passadas em um formulário ficam contidas dentro do corpo da requisição:
  - `req.body`.
- Se queremos recuperar informações do atributo **nome**, utilizamos a instrução:
  - `req.body.nome`
  - O valor da propriedade **name** no `form` no lado do cliente é o mesmo utilizado no lado do servidor

Exemplo

- Suponha um servidor que recebe a idade de uma pessoa e diz se ela é maior de idade ou não:

Exemplo

- Suponha um servidor que recebe a idade de uma pessoa e diz se ela é maior de idade ou não:

```
app.post('/validaridade', function(req, res){
    if (req.body.idade < 18) {
        res.send("<p>Você émenor de idade</p>");
    }else {
        res.send("<p>Você émaior de idade</p>");
    }
});
```

Introducao ao Node.JS

Exercícios

Exercício 1

- Crie um formulário contendo **nome**, **e-mail** e **ano de nascimento**.
- O servidor deve devolver:
  - Se você nasceu em um ano bissexto.
    - anos bissextos ocorrem a cada quatro anos (exceto anos múltiplos de 100 que não são múltiplos de 400).
  - Se você tem idade para votar (maior de 16 anos).
  - Se você tem idade para dirigir (maior de 18 anos).

Referências

Paul, S. (2020).
Read html form data using get and post method in node.js.
https://medium.com/swlh/read-html-form-data-using-get-and-post-method-in-node-js-8d2c7880adbf.

Obrigado

gustavo.custodio@ulife.com.br