Workshop NodeJS

Nível Básico

18/02/2017

Agenda

09:15 - 09:20	Sobre vocês
09:20 - 09:22	Sobre nós
09:22 - 09:40	O protocolo HTTP
09:40 - 10:00	MongoDB
10:00 - 10:30	NodeJS
10:30 - 12:00	Hands on: Criando um blog com NodeJS
12:00 - 13:00	Dúvidas

Sobre vocês

40 segundos para cada um falar

Sobre vocês

Contem um pouco sobre vocês!

1. Nome
2. O que faz da vida
3. Experiência com NodeJS e MongoDB
4. O que espera desse Workshop

Sobre nós

Sobre nós / Sócios

Guilherme Uezima

Graduado em Sistemas de Informação pelo Mackenzie, atua há mais de cinco anos com o desenvolvimento de soluções tecnológicas para diversas plataformas.

Gustavo Soré

Graduado em Sistemas de Informação no Mackenzie. Desenvolvedor de aplicativos, especialista em iOS. Responsável por aplicativos de CRM e mídia digital para o setor farmacêutico na América Latina.

Michel Zarzour F.

Graduado em Sistemas de Informação pelo Mackenzie, possui grande experiência na área de desenvolvimento de aplicações WebApp e participação em grupos de startup.

Otávio R. Rossi

Graduado em Sistemas de Informação pelo Mackenzie, possui grande experiência na área de desenvolvimento de aplicações Web e consultoria de negócios na área de tecnologia em grandes empresas do mercado

Sobre nós

Tecnologias e ferramentas modernas para desenvolvimento de soluções em cloud

Gerenciamento dos feedbacks, recomendações de mercado e implementação de melhorias

Foco na experiência do usuário na utilização das soluções customizadas em multiplataformas

http://inopus.com.br

HTTP

Introdução

HTTP / Introdução

Hypertext Transfer Protocol

"Is an application-level protocol for distributed, collaborative, hypermedia information systems. HTTP has been in use by the World-Wide Web global information initiative since 1990. The first version of HTTP, referred to as HTTP/0.9, was a simple protocol for raw data transfer across the Internet".

IETF

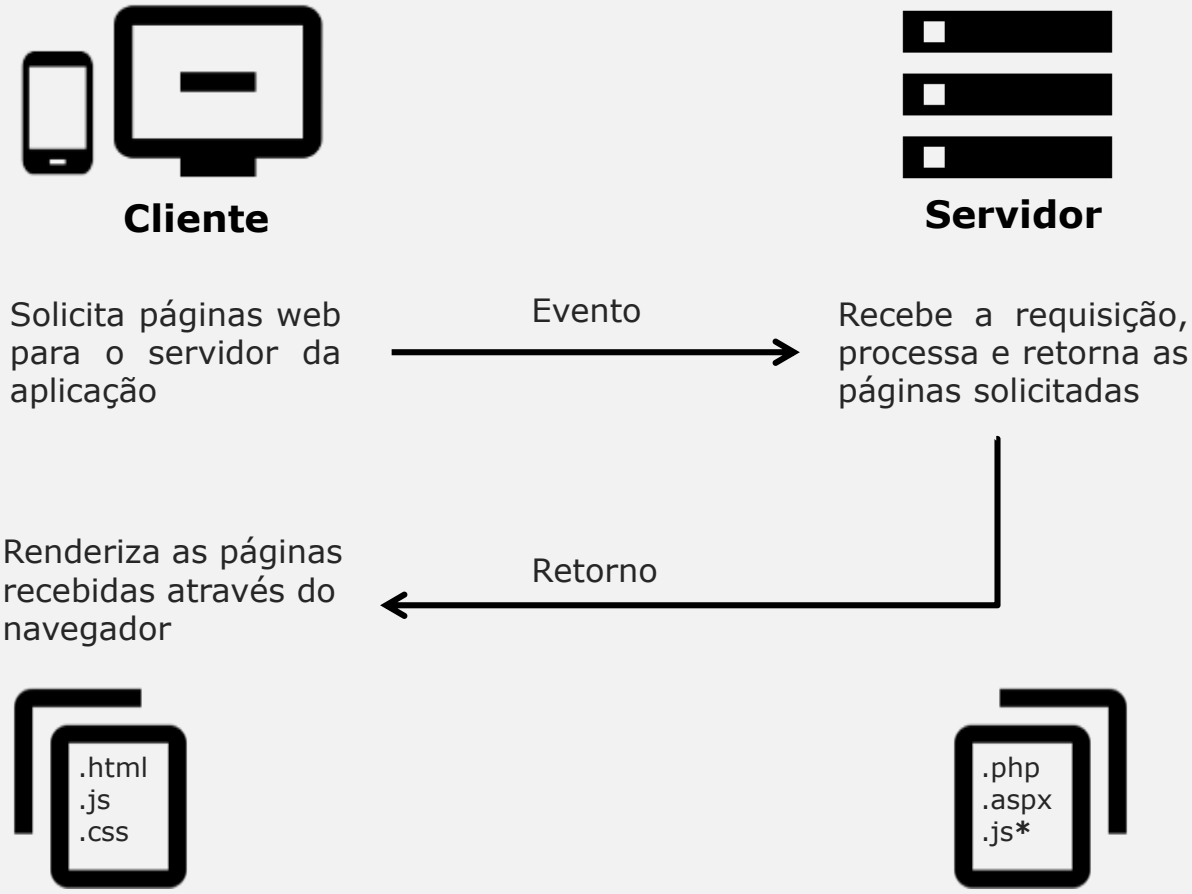
- Camada de aplicação
- *Hypermedia*
- Usado pela WWW (*World-Wide Web*)
- Atualmente da versão 2.0 (HTTP/2)

Utilização:

HTTP

Como funciona em relação as aplicações web

HTTP / Como funciona em relação as aplicações web



Métodos HTTP

GET, POST, HEAD...

Métodos HTTP / GET, POST, HEAD...

O HTTP possuí alguns métodos que indicam a ação requerida com aquela requisição. Dentre esses diferentes tipos de métodos existentes, existem 5 que são os mais utilizados e difundidos.

Method	*Request Body*	*Response Body*
GET	No	Yes
HEAD	No	No
POST	Yes	Yes
PUT	Yes	Yes
DELETE	No	Yes

Métodos HTTP / GET, POST, HEAD...

1

GET https://www.google.com.br/search?q=tecnodrom&ie=utf-8&oe=utf-8&client=firefox-b-ab

2

GET http://tecnodrom.com/Postagens/380/inopus-vence-hackathon-da-ibm

3

POST https://rapordo.com/usuario/entrar

usuario: otavio
senha: af12354abcd134fed

Tipos de requisições HTTP

Formulários e AJAX

Tipos de requisições HTTP / Formulários e Ajax

Formulários

1. `<form>` no HTML representa uma seção do documento que contém elementos interativos para serem enviados ao servidor;

2. Principais atributos do `<form>`:
   2.1. `action` – URL que receberá o formulário;
   2.2. `enctype` – utilizado no POST, *MIME Type* do conteúdo;
   2.3. `method` – Método HTTP que será utilizado (GET ou POST).

Ajax

1. Asynchronous JavaScript + XML;

2. Termo utilizado para descrever uma "nova" (2005) abordagem na utilização de diversas tecnologias;

3. XMLHttpRequest();
   3.1. API para transferência de dados;
   3.2. Modos assíncronos e síncronos;
   3.3. Utilizado principalmente para JSON.

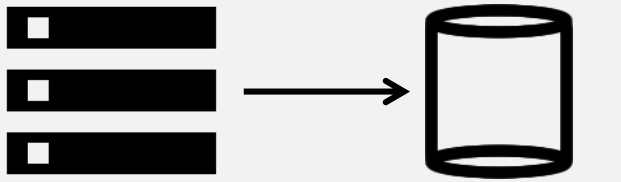
MongoDB

Introdução

MongoDB / Introdução

MongoDB

Mongo é um banco de dados não relacional (NoSQL), de plataforma aberta, orientado a documentos, cuja escalabilidade é mais barata e menos trabalhosa, visto que não exige uma máquina extremamente poderosa para o processamento de um grande volume de dados.



Exemplo de coleção do MongoDB

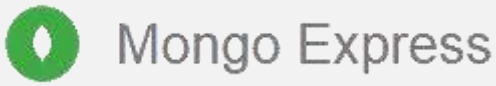
Collection `Pessoa`

```
{
   "_id": ObjectID("58a25b672883d22f94683d1b"),
   "nome": "João da Silva",
   "idade": 28,
   "profissão": "Desenvolvedor(a)"
}
{
   "_id": ObjectID("58a25b772883d22f94683d1c"),
   "nome": "Maria da Silva",
   "idade": 27,
   "profissão": "Desenvolvedor(a)"
}
```

MongoDB

Mongo Express

MongoDB / Conectando o Mongo Express



"Web-based MongoDB admin interface written with Node.js, Express and Bootstrap3"

Features:

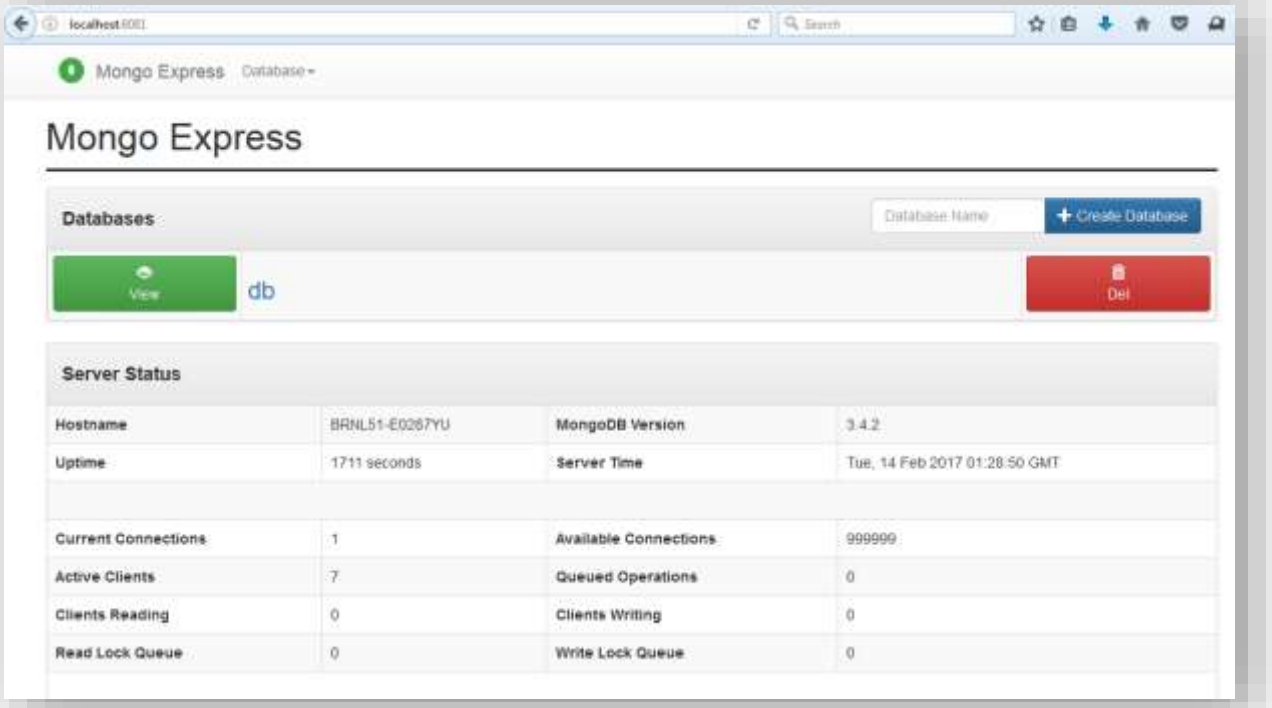
- Connect to multiple databases
- View/add/delete databases
- View/add/rename/delete collections
- View/add/update/delete documents
- Preview audio/video/image assets inline in collection view
- Nested and/or large objects are collapsible for easy overview
- Async on-demand loading of big document properties (>100KB default) to keep collection view fast
- GridFS support - add/get/delete incredibly large files
- Use BSON data types in documents
- Mobile / Responsive - Bootstrap 3 works passably on small screens when you're in a bind
- Connect and authenticate to individual databases
- Authenticate as admin to view all databases
- Database blacklist/whitelist
- Custom CA and CA validation disabling
- Supports replica sets

Github mongo-express

MongoDB / Conectando o Mongo Express

MongoDB / Conectando o Mongo Express

- ✓ Faça download do projeto em https://github.com/mongo-express/mongo-express
- ✓ Coloque todos os arquivos em uma pasta.
- ✓ Acesse a pasta via CMD e execute o comando `node app.js`.
- ✓ O usuário e senha do painel do Mongo Express são `admin` e `pass`, respectivamente.



- ✓ Caso o seu banco de dados necessite de um usuário e senha para conexão, o mesmo deve ser definido no arquivo `config.default.js`.

NodeJS

Hello World

NodeJS / Hello World

Para criar o seu projeto NodeJS, basta acessar a pasta que deseja colocar o projeto e, através do CMD, utilizar o comando `npm init`.

Esse comando criará um arquivo `package.json` no diretório.

package.json

```
{
  "name": "zzzzzzz",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

NodeJS / Hello World

app.js

```
1  #!/bin/env node
2
3  var express = require('express');
4  var app = express();
5
6  app.set('view engine', 'ejs');
7
8  app.get('/hello_world', function (req, res) {
9      res.render('paginas/hello_world');
10 });
11
12 var porta = 8080;
13 app.listen(porta);
14 console.log("***** Servidor rodando na porta " + porta + " as " + new Date().toISOString()
   + " *****");
15
```

NodeJS / Hello World

views/paginas/hello_world.ejs

```
1 <h1>
2   <%="Hello World"%>
3   </h1>
4
```

NodeJS / Hello World

Para iniciar o servidor NodeJS na sua máquina, basta abrir o CMD e digitar o comando `node app.js`.

NodeJS / Hello World

Para iniciar o servidor NodeJS na sua máquina, basta abrir o CMD e digitar o comando `node app.js`.

CMD

```
\workshop-nodejs-20170218>node app.js
module.js:471
    throw err;
    ^

Error: Cannot find module 'express'
    at Function.Module._resolveFilename (module.js:469:15)
    at Function.Module._load (module.js:417:25)
    at Module.require (module.js:497:17)
    at require (internal/module.js:20:19)
    at Object.<anonymous> (C:\Users\Otaviorossi\OneDrive - Inopus\PC\inopus\inop
us\workshop\20170218\github\workshop-nodejs-20170218\app.js:3:15)
    at Module._compile (module.js:570:32)
    at Object.Module._extensions..js (module.js:579:10)
    at Module.load (module.js:487:32)
    at tryModuleLoad (module.js:446:12)
    at Function.Module._load (module.js:438:3)
```

NodeJS / Hello World

Para iniciar o servidor NodeJS na sua máquina, basta abrir o CMD e digitar o comando `node app.js`.

CMD

```
\workshop-nodejs-20170218>node app.js
module.js:471
    throw err;
    ^

Error: Cannot find module 'express'
    at Function.Module._resolveFilename (module.js:469:15)
    at Function.Module._load (module.js:417:25)
    at Module.require (module.js:497:17)
    at require (internal/module.js:20:19)
    at Object.<anonymous> (C:\Users\Otaviorossi\OneDrive - Inopus\PC\inopus\inop
us\workshop\20170218\github\workshop-nodejs-20170218\app.js:3:15)
    at Module._compile (module.js:570:32)
    at Object.Module._extensions..js (module.js:579:10)
    at Module.load (module.js:487:32)
    at tryModuleLoad (module.js:446:12)
    at Function.Module._load (module.js:438:3)
```

Para liberar a instalação dos módulos do NodeJS, utilize os comandos:

```
npm config set proxy http://172.16.0.10:3128
npm config set http-proxy http://172.16.0.10:3128
```

Utilize o comando `npm install express --save` para instalar o módulo `express`.

Blog

Nova postagem

Blog / Nova postagem

views/partes/cabecalho.ejs

```
<!doctype html>
<html>
<head>
    <title>Blog do Uezima - Workshop Mackenzie</title>
    <meta charset="utf-8">
    <meta name="author" title="Guilherme Uezima, Otávio R. Rossi">
    <link rel="author" href="http://otavio.rrossi.eti.br" title="Otávio R. Rossi">
    <link rel="author" href="https://www.linkedin.com/in/guilherme-uezima-a2931a65"
    title="Guilherme Uezima">
    <link rel="icon" href="/img/uezima.png">
    <link href="/css/geral.css" rel="stylesheet" type="text/css" media="all">
    <link href="/css/materialize.min.css" rel="stylesheet" type="text/css" media="all">
    <script type="text/javascript" src="/js/jquery-3.1.1.min.js"></script>
    <script type="text/javascript" src="/js/materialize.min.js"></script>
</head>
<body>
    <header>
        <nav class="light-blue lighten-1" role="navigation">
            <div class="nav-wrapper container"><a id="logo-container" href="/"
            class="brand-logo">Blog do Uezima</a>
                <ul class="right hide-on-med-and-down">
                    <li><a href="http://inopus.com.br" target="_blank">Inopus</a></li>
                </ul>
            </div>
        </nav>
    </header>
    <div id="tudo" class="container">
        <div class="section">
```

Blog / Nova postagem

Links para download dos arquivos JS e CSS necessários para esse projeto.

https://github.com/inopus/workshop-nodejs-20170218/raw/master/views/css.zip

https://github.com/inopus/workshop-nodejs-20170218/raw/master/views/fonts.zip

https://github.com/inopus/workshop-nodejs-20170218/raw/master/views/js.zip

Blog / Nova postagem

views/partes/rodape.ejs

```
</div>
</div>
</body>
<footer class="page-footer orange">
    <div class="container">
        <div class="row">
            <div class="col l6 s12">
                <h5 class="white-text">&copy; 2017 Blog do Uezima</h5>
                <p class="grey-text text-lighten-4">Parabéns! Você está criando sua
                primeira aplicação em NodeJS :)</p>
            </div>
            <div class="col l3 s12">
                <h5 class="white-text">Páginas</h5>
                <ul>
                    <li><a class="white-text" href="/postar">Criar uma nova postagem</a>
                    </li>
                </ul>
            </div>
            <div class="col l3 s12">
                <h5 class="white-text">Contatos</h5>
                <ul>
                    <li><a class="white-text" href="http://inopus.com.br">Inopus</a></li>
                    <li><a class="white-text"
                    href="http://fb.com/inopussolucoes">fb.com/inopussolucoes</a></li>
                </ul>
            </div>
        </div>
    </div>
</footer>
```

Blog / Nova postagem

views/paginas/postar.ejs

```
1  <% include ../partes/cabecalho %>
2      <form method="post" action="/postar/salvar">
3          <div class="campo">
4              <label for="titulo">Título</label>
5              <input type="text" name="titulo" id="titulo" method="get">
6          </div>
7          <div class="campo">
8              <label for="categoria">Categoria</label>
9              <select name="categoria">
10                 <option value="">-- Selecione uma opção --</option>
11                 <option value="Jogos">Jogos</option>
12                 <option value="Notícias">Notícias</option>
13                 <option value="Aleatórios">Aleatórios</option>
14             </select>
15         </div>
16         <div class="campo">
17             <label for="postagem">Postagem</label>
18             <textarea name="postagem" id="postagem"></textarea>
19         </div>
20         <div class="campo">
21             <button>Gravar</button>
22         </div>
23     </form>
24     <% include ../partes/rodape %>
```

Blog / Nova postagem

app.js

...

```
12  var bodyParser = require('body-parser');
13
14  app.use(bodyParser.urlencoded({
15      extended: false
16  }));
```

...

```
24  app.get('/postar', function (req, res) {
25      res.render('paginas/postar');
26  });
```

...

Blog / Nova postagem

app.js

```
...
30 app.post('/postar/salvar', function (req, res) {
31     var clt = dbCon[0].collection("Postagem");
32     var postagem = {
33         titulo: req.body.titulo,
34         categoria: req.body.categoria,
35         postagem: req.body.postagem
36     };
37     clt.insert(postagem, {
38         w: 1
39     }, function (err, result) {
40         if (err) {
41             console.log("*** Erro: ");
42             console.log(err);
43             console.log("***");
44             res.redirect("/?msg=erro");
45         } else {
46             res.redirect("/?msg=sucesso");
47         }
48     });
49 });
...
```

Blog

Lista de postagens

Blog / Lista de postagens

app.js

...

```
69 app.get('/', function (req, res) {
70     var clt = dbCon[0].collection("Postagem");
71     clt.find().toArray(function (err, results) {
72         if (err) {
73             console.log("*** Erro: ");
74             console.log(err);
75             console.log("***");
76             res.redirect("/?msg=erro");
77         } else {
78             console.log(results);
79             res.render('paginas/index', {
80                 msg: req.query.msg,
81                 postagens: results
82             });
83         }
84     });
85 });
```

Blog / Lista de postagens

views/paginas/index.ejs

```
<% include ../partes/cabecalho %>
    <div class="row">
        <% if(postagens.length > 0) {
            for (var i = 0; i < postagens.length; i++) { %>
            <div class="col s12">
                <div class="card postagem" style="cursor: pointer;"
                onclick="window.location.href='/detalhe/<%=postagens[i]._id%>'">
                    <p class="titulo-postagem">
                        <%=postagens[i].titulo%>
                    </p>
                    <p class="categoria-postagem">
                        <%=postagens[i].categoria%>
                    </p>
                </div>
            </div>
            <%
            }
        } else {
            %>
                <div class="col s12">
                    <div class="card postagem">
                        <p class="conteudo-postagem">
                            (Não há postagens disponíveis)
                        </p>
                    </div>
                </div>
                <%
        }
        %>
    </div>
    <% include ../partes/rodape %>
```

Blog

Detalhe da postagem

Blog / Detalhe da postagem

app.js

```
...
```

```
49 app.get('/detalhe/:id', function (req, res) {
50     var clt = dbCon[0].collection("Postagem");
51     clt.findOne({
52         '_id': new mongodb.ObjectID(req.params.id)
53     }, function (err, result) {
54         if (err) {
55             console.log("*** Erro: ");
56             console.log(err);
57             console.log("***");
58             res.redirect("/?msg=erro");
59         } else {
60             res.render('paginas/detalhe', {
61                 titulo: result.titulo,
62                 categoria: result.categoria,
63                 postagem: result.postagem
64             });
65         }
66     });
67 });
```

Blog / Detalhe da postagem

views/paginas/detalhe.ejs

```
<% include ../partes/cabecalho %>
    <% if(titulo && categoria && postagem) {
    %>
        <div class="row">
            <div class="col s12">
                <div class="card postagem">
                    <p class="titulo-postagem">
                        <%=titulo%>
                    </p>
                    <p class="categoria-postagem">
                        <%=categoria%>
                    </p>
                    <p class="conteudo-postagem">
                        <%=postagem%>
                    </p>
                </div>
            </div>
        </div>
        <%
    } else {
        %>
            <meta http-equiv="refresh" content="0;URL=/">
            <%
    }
    %>
                <% include ../partes/rodape %>
```

Blog

Arquivos estáticos (JS, CSS...)

Blog / Arquivos estáticos (JS, CSS...)

Como todas as requisições feitas ao nosso servidor NodeJS devem possuem uma `function` especifica em uma rota, será necessário especificar que possuímos uma pasta com arquivos que podem ser acessados diretamente, pois possuem conteúdos estáticos (como por exemplo, imagens, JSs e CSSs), não sendo necessário que o Node processe através de uma rota com uma `function` esses conteúdos.

app.js

```
...
 9
10  app.use(express.static('./views'));
11
...
```

Blog

package.json

Blog / package.json

Agora, todos os módulos que foram adicionados estão especificados nas dependências do seu projeto.

package.json

```
{
    "name": "zzzzzzz",
    "version": "1.0.0",
    "description": "",
    "main": "app.js",
    "scripts": {
        "test": "echo \"Error: no test specified\" && exit 1"
    },
    "author": "",
    "license": "ISC",
    "dependencies": {
        "body-parser": "*",
        "ejs": "*",
        "express": "*",
        "mongodb": "*"
    }
}
```

Blog / package.json

Agora, todos os módulos que foram adicionados estão especificados nas dependências do seu projeto.

package.json

```
{
    "name": "zzzzzzz",
    "version": "1.0.0",
    "description": "",
    "main": "app.js",
    "scripts": {
        "test": "echo \"Error: no test specified\" && exit 1"
    },
    "author": "",
    "license": "ISC",
    "dependencies": {
        "body-parser": "*",
        "ejs": "*",
        "express": "*",
        "mongodb": "*"
    }
}
```

Você poderia, no começo do projeto, ter adicionado manualmente as dependências nesse arquivo e utilizado o comando `npm install` para instalar todas elas de uma única vez.

Blog / package.json

Agora, todos os módulos que foram adicionados estão especificados nas dependências do seu projeto.

package.json

```
{
    "name": "zzzzzzz",
    "version": "1.0.0",
    "description": "",
    "main": "app.js",
    "scripts": {
        "test": "echo \"Error: no test specified\" && exit 1"
    },
    "author": "",
    "license": "ISC",
    "dependencies": {
        "body-parser": "*",
        "ejs": "*",
        "express": "*",
        "mongodb": "*"
    }
}
```

O `*` indica que será utilizada sempre a última versão disponível, quanto o `x.x` significa que utilizaremos especificamente uma versão

Você poderia, no começo do projeto, ter adicionado manualmente as dependências nesse arquivo e utilizado o comando `npm install` para instalar todas elas de uma única vez.

INOPUS

Obrigado!

Guilherme Uezima
Otávio R. Rossi

Contatos

Guilherme Uezima

Design e Integração
+ 55 11 9 7692-7785
guilherme@inopus.com.br

Gustavo Soré

Desenvolvimento Back-end
+ 55 11 9 8585-4328
gustavo@inopus.com.br

Michel Zarzour Filho

Desenvolvimento Front-end
+ 55 11 9 8174-7375
michel@inopus.com.br

Otávio R. Rossi

Consultor de Negócios
+ 55 11 9 8544-4724
otavio@inopus.com.br

Referências

- https://github.com/mongo-express/mongo-express
- https://scotch.io/tutorials/use-ejs-to-template-your-node-application
- https://docs.mongodb.com/manual/installation/
- http://expressjs.com/
- https://msdn.microsoft.com/en-us/library/ms526971(v=exchg.10).aspx
- https://msdn.microsoft.com/en-us/library/ms527355(v=exchg.10).aspx
- https://developer.mozilla.org/en/docs/AJAX
- https://tools.ietf.org/html/rfc7540
- https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form
- https://www.shimmercat.com/en/blog/articles/whats-push/
- http://stackoverflow.com/questions/2404742/how-to-install-mongodb-on-windows
- https://www.shimmercat.com/en/blog/articles/whats-push/
- https://en.wikipedia.org/wiki/HTTP/2
- http://materializecss.com/getting-started.html