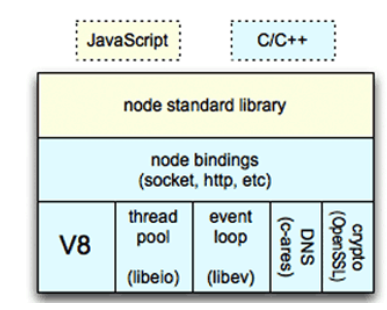
**Node.Js**

Não é um framework. É uma runtime, ou seja, um set de bibliotecas escritas em JavaScript, que possibilitam a execução de JavaScript no servidor. Uma tecnologia server-side.

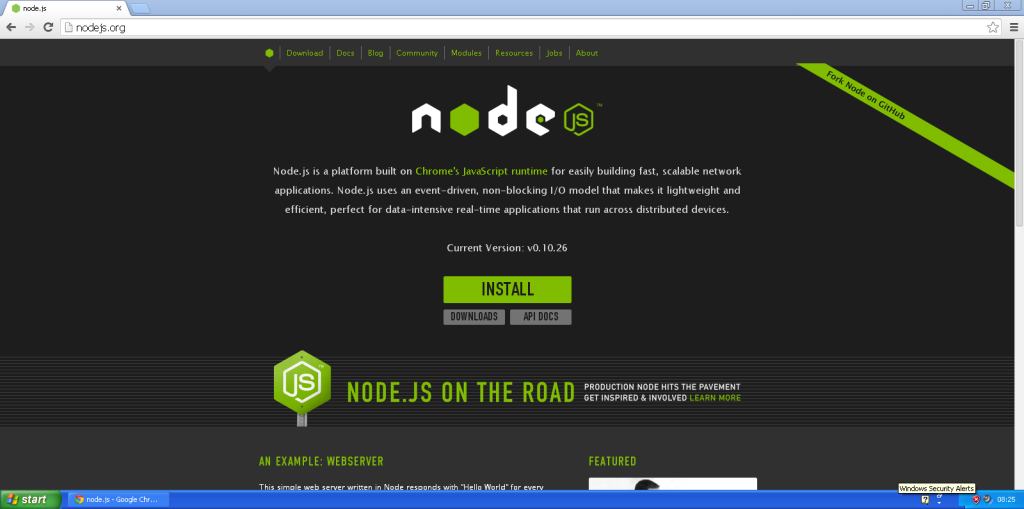


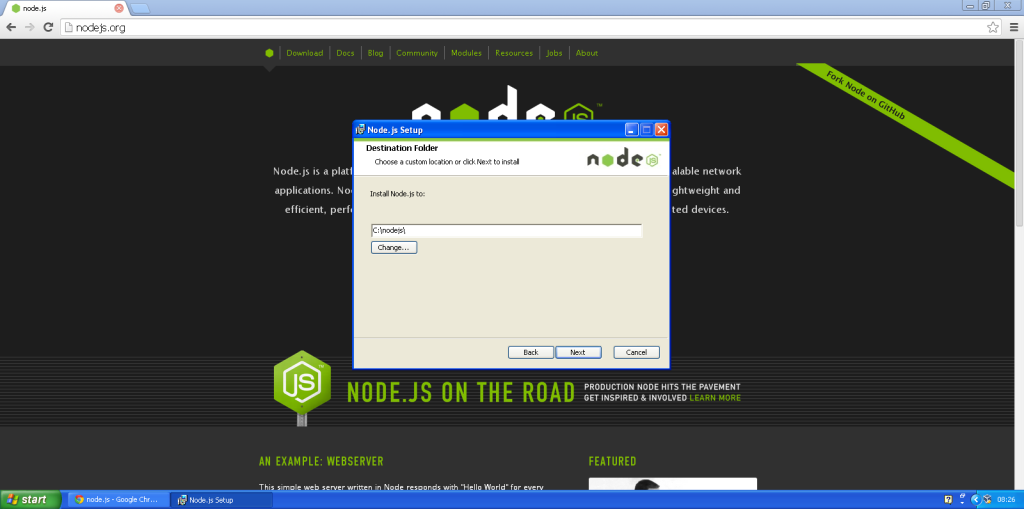
Nodejs,org,

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

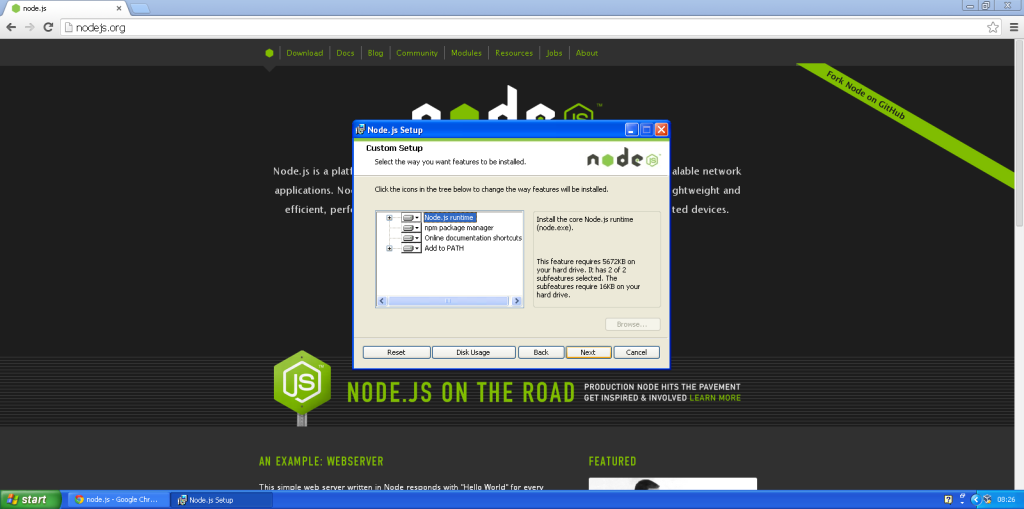
Instalando

Baixe e instale no servidor a versão mais recente disponível para Windows no [site oficial do NodeJS](http://www.nodejs.org/).

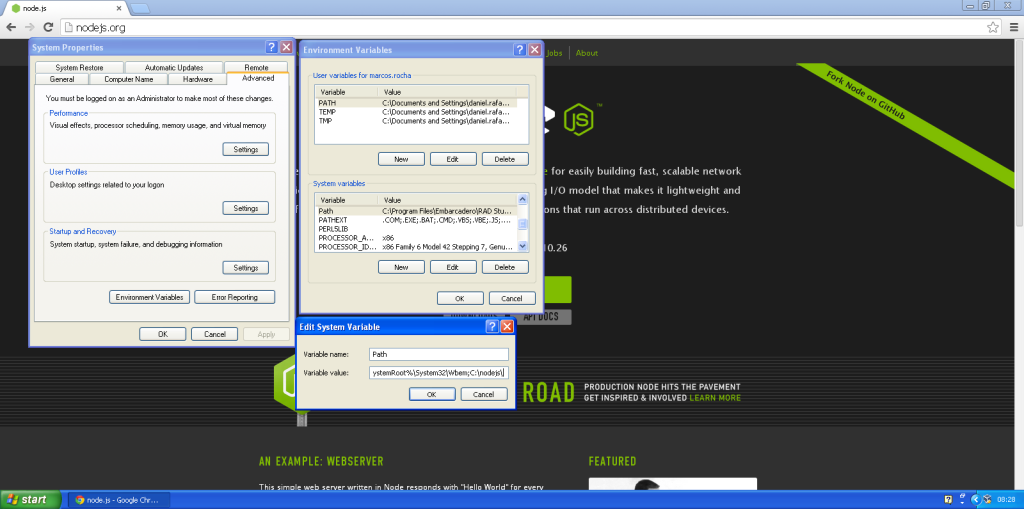
Acesse a página do [Node.js](http://www.nodejs.org/) e clique no botão “INSTALL”. Ele fará seu navegador iniciar o download da versão mais atualizada do Node.js. Se preferir utilizar uma versão mais antiga, o botão DOWNLOADS irá levar à página de downloads onde é possível escolher o sistema operacional, plataforma e etc.  
[](http://marcosrocha.net/wp-content/uploads/2015/04/NodeJS-01-P%C3%A1gina-do-Node.png)

**2. Instalação do Node.js**  
Após feito o download da instalação do Node, abra o aplicativo e aceite os termos de uso (hehehe), e na janela para selecionar a pasta, coloque C:\nodejs\ ou outra unidade de sua preferência. A grande sacada é, **garanta que não haja espaço no nome do diretório**, pois caso contrário você poderá ter problemas utilizando o Node.js ou suas aplicações. Tenha em mente que o Node assim como muitos aplicativos são construídos para funcionar em sistemas baseados em Unix/Linux e funcionam no Windows por Cygwin.  
[](http://marcosrocha.net/wp-content/uploads/2015/04/NodeJS-02-Pasta-do-Node.png)

Na tela de seleção de pacotes, garanta que **TODOS** estejam selecionados, principalmente o Node.js runtime e o NPM. Falaremos sobre o NPM mais tarde. Depois prossiga com a instalação até o fim com “Next, Next, Finish”.

[](http://marcosrocha.net/wp-content/uploads/2015/04/NodeJS-03-Pacotes-do-Node.png)

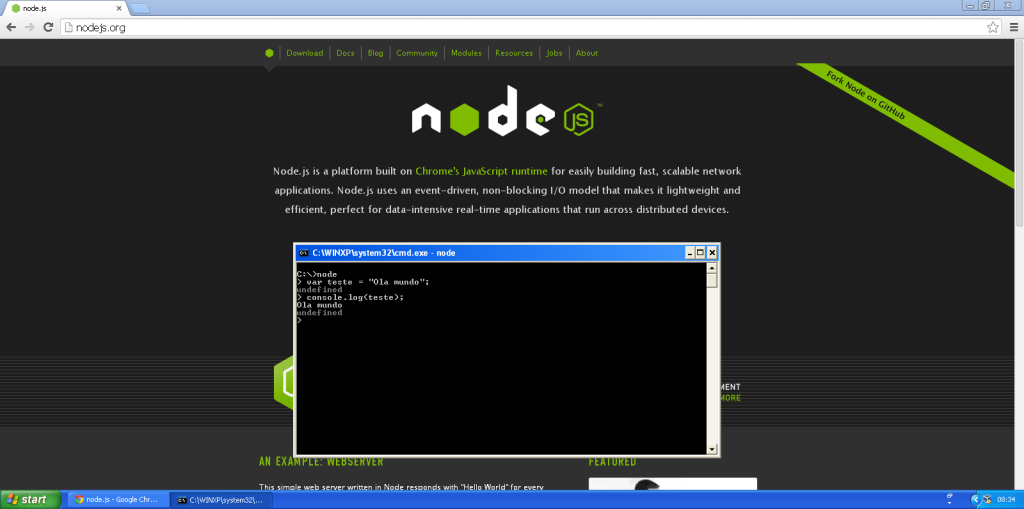
Veja que na variável PATH de ambiente do windows, foi incluído o caminho do Node.js que você escolheu. O ideal é que o caminho do Node.js seja o primeiro da lista.

[](http://marcosrocha.net/wp-content/uploads/2015/04/NodeJS-04-System-Path.png)

Agora o Node.js já está instalado, você deve conseguir utiliza-lo no “Prompt de Comando” do Windows, por exemplo em nosso código de teste digite uma linha, dê enter, digite a próxima e enter.

var teste = "Ola Mundo";  
console.log(teste);

Deve aparecer um resultado semelhante a este:

[](http://marcosrocha.net/wp-content/uploads/2015/04/NodeJS-06-Node.png)

3. Depois de instalado, vá novamente no Executar/Run e digite:

|  |  |
| --- | --- |
|  | node -v |

Isso deve lhe devolver a versão do NodeJS instalada no servidor. Se não funcionar, reinicie o servidor e tente novamente.

Publique a sua aplicação NodeJS em uma pasta do servidor. Minha sugestão é C:\domains\nomedoapp\www.

Abra uma janela do CMD com permissão de administrador e navegue usando cd até a pasta da sua aplicação Node.js. Exemplo:

|  |  |
| --- | --- |
| 1 | cd C:\domains\nomedoapp\www |

Uma vez lá, digite o seguinte comando para instalar as dependências do seu projeto:

|  |  |
| --- | --- |
| 1 | npm install |

Isso deverá ser o suficiente, mas se quiser ter certeza de que tudo que necessita para rodar Node.js está instalado em seu servidor, execute também os seguintes comandos em sequência:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | npm install express -g  npm install url -g  npm install fresh -g  npm install cookie -g  npm install methods -g  npm install crc -g  npm install send -g  npm install connect -g  npm install commander -g |

Agora finalmente execute seu app Node.js com o comando abaixo ainda “dentro” da pasta do seu projeto (ou navegue novamente usando cd):

|  |  |
| --- | --- |
| 1 | npm start |

### Passo 6: Instalar o Windows Service

Por padrão, o seu site Node.js ficará executando enquanto a janela do console que iniciou seu site estiver funcionando. Se você fechar ela, ou seu servidor reiniciar (malditos Windows Updates!) já era, seu site vai ficar fora do ar até que você execute o npm start de novo.

Para que isso não aconteça, você deve instalar seu site como um Windows Service. Para fazer isso, primeiro instale o [módulo node-windows](https://github.com/coreybutler/node-windows)globalmente:

|  |  |
| --- | --- |
| 1 | npm install -g node-windows |

Agora rode o seguinte comando (dentro da pasta do seu projeto) para incluir uma referência deste módulo ao seu projeto:

|  |  |
| --- | --- |
| 1 | npm link node-windows |

Depois, dentro do seu projeto Node.js (na raiz mesmo) crie um arquivo service.js com o seguinte conteúdo:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | var Service = require('node-windows').Service;    // Create a new service object  var svc = new Service({    name:'Nome da sua Aplicação',    description: 'Apenas uma descrição',    script: 'C:\\domains\\sitenodejs\\bin\\www'  });    // Listen for the "install" event, which indicates the  // process is available as a service.  svc.on('install',function(){    svc.start();  });    svc.install(); |

Troque as propriedades name e description de acordo com seu gosto, mas atente à propriedade script nesse código, que deve conter o caminho absoluto até o arquivo JS que inicia sua aplicação. No meu caso, como estou usando express, estou apontando para o arquivo www que fica na pasta bin do projeto (curiosamente ele não possui extensão, mas é um arquivo).

Se você fez tudo corretamente, vá até Ferramentas Administrativas > Serviços (Administrative Tools > Services ou services.msc no Run/Executar do Windows) e seu serviço vai aparecer lá com o name que definiu ali no script, lhe permitindo alterar suas configurações de inicialização, dar Start, Stop, etc.

Outra coisa bacana desse módulo é que ele manda todas as mensagens para o Event Viewer/Visualizador de Eventos do Windows. Assim, caso sua aplicação Node.js não esteja funcionando ou seu serviço não esteja inicializando, basta dar uma olhada lá, na seção Windows > Applications para entender o que está acontecendo.

Caso precise remover esse serviço (para instalar uma versão mais atualizada, por exemplo) rode o comando abaixo no cmd:

|  |  |
| --- | --- |
| 1  2 | SC STOP servicename  SC DELETE servicename |

Esse servicename você encontra nas propriedades do Windows Service que deseja excluir.

### Bônus: PM2

PM2 é um projeto open-source criado e mantido pela empresa Keymetrics.io, que além do PM2 (que é gratuito) vende um serviço de gerenciamento de performance de aplicações Node.js homônimo. Só para você ter uma ideia do que o PM2 é hoje, são mais de 20M de downloads e empresas como IBM, Microsoft e PayPal usando, o que o torna, disparado, a melhor solução de process manager pra Node, muito mais do que seus principais concorrentes, o Forever e o Nodemon.

Pra usar o PM2 é muito simples, primeiro instale globalmente o módulo do PM2:

|  |  |
| --- | --- |
| 1 | $ npm install pm2 -g |

Depois, quando quiser iniciar o processo da sua aplicação Node.js:

|  |  |
| --- | --- |
| 1 | $ pm2 start myApp.js |

Executando

|  |
| --- |
| node |

Exemplo de commando por console:

console.log**(**'Hello World'**)**;

Para encerrar:

CTRL ^ C (2 vezes)

ou

.exit

## Primeiros passos com Node.JS

## **Introdução**

Node.JS é um interpretador de javascript que roda do lado do servidor. Pensado para trabalhar como um servidor web, ele foi criado como uma plataforma sobre o runtime do interpretador V8 do Google Chrome.

Isso é bom porque o V8 transforma o javascript em código nativo de maquina, o que poderia ser traduzido como performance. A idéia central é escrever aplicações em rede rápidas e escaláveis através de chamadas assíncronas.

Muito interessante, não? Mas por que escrever um monte de código para disponibilizar um servidor web, se já podemos fazer isso com um Apache Tomcat ou qualquer outro servidor? Bem, esta pergunta nos traz ao presente texto.

(Para concluir este artigo com sucesso é necessário ter o node instalado e configurado na máquina. Maiores informações [aqui.](http://www.nodejs.org/))

Criando um servidor Node simples:

Arquivo: server.js

var http = require('http');

http.createServer(function(req, res) {

res.writeHead(200, {

'Content-Type': 'text/plain'

});

res.end('Hello World\n');

}).listen(1337, '127.0.0.1');

console.log('Server running at <http://127.0.0.1:1337/>');

Executar:

> node server.js

Pronto! Agora temos o servidor rodando localmente. Basta acessar a URL informada para ver a aplicação respondendo a sua primeira requisição ao servidor node.

Criar a pasta ‘public’ e o arquivo index.html dentro da mesma.

<!doctype html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Hello</title>

</head>

<body>

<p>Hello world!</p>

</body>

</html>