NodeJS

Prof. Rodrigo Macedo

Escopo do Curso

- Contextualização Desenvolvimento Web.
- Características NodeJS.
- Configurações NodeJS.
- Componentes NodeJS.
- NPM.
- Questões de concursos

Contextualização

- Javascript é uma das três linguagens mais importante no desenvolvimento web.
- São elas:

1. HTML para definir o conteúdo de uma página web.
2. CSS para especificar o layout de uma página web.
3. Javascript para desenvolver o comportamento de uma página web.

Com Javascript, podemos interagir entre os componentes de uma página web por meio do DOM (Document Object Model).

NodeJS

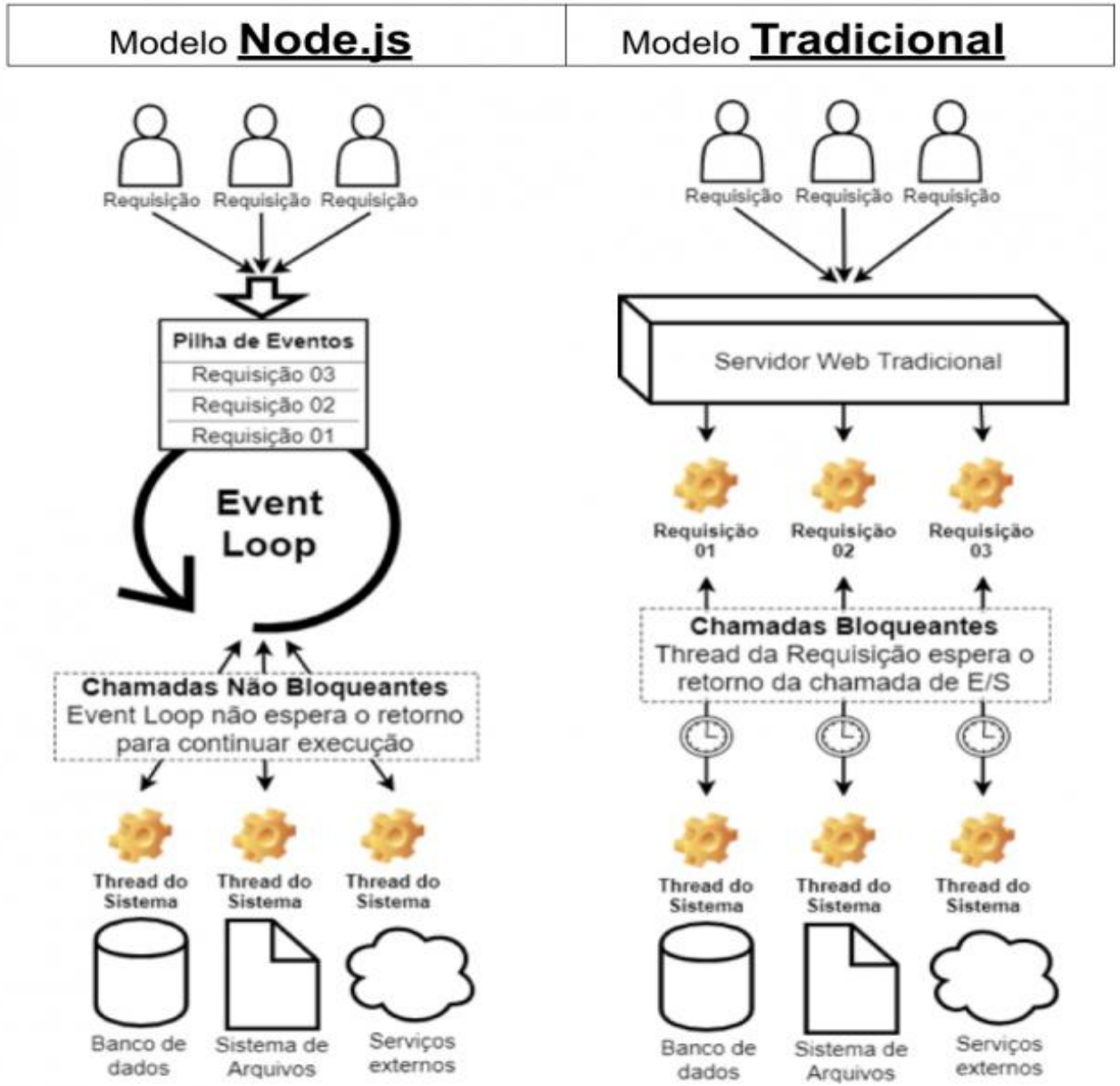
- Node.js é um ambiente de execução de código JavaScript no lado do servidor, open-source e multiplataforma.
- Node.js permite que possamos usar JavaScript como uma linguagem de scripting server-side também, permitindo criar conteúdo web dinâmico antes da página aparecer no navegador do usuário.
- Assim, Node.js se tornou um dos elementos fundamentais do paradigma “fullstack” JavaScript, permitindo que todas as camadas de um projeto possam ser desenvolvidas usando apenas essa linguagem.

NodeJS

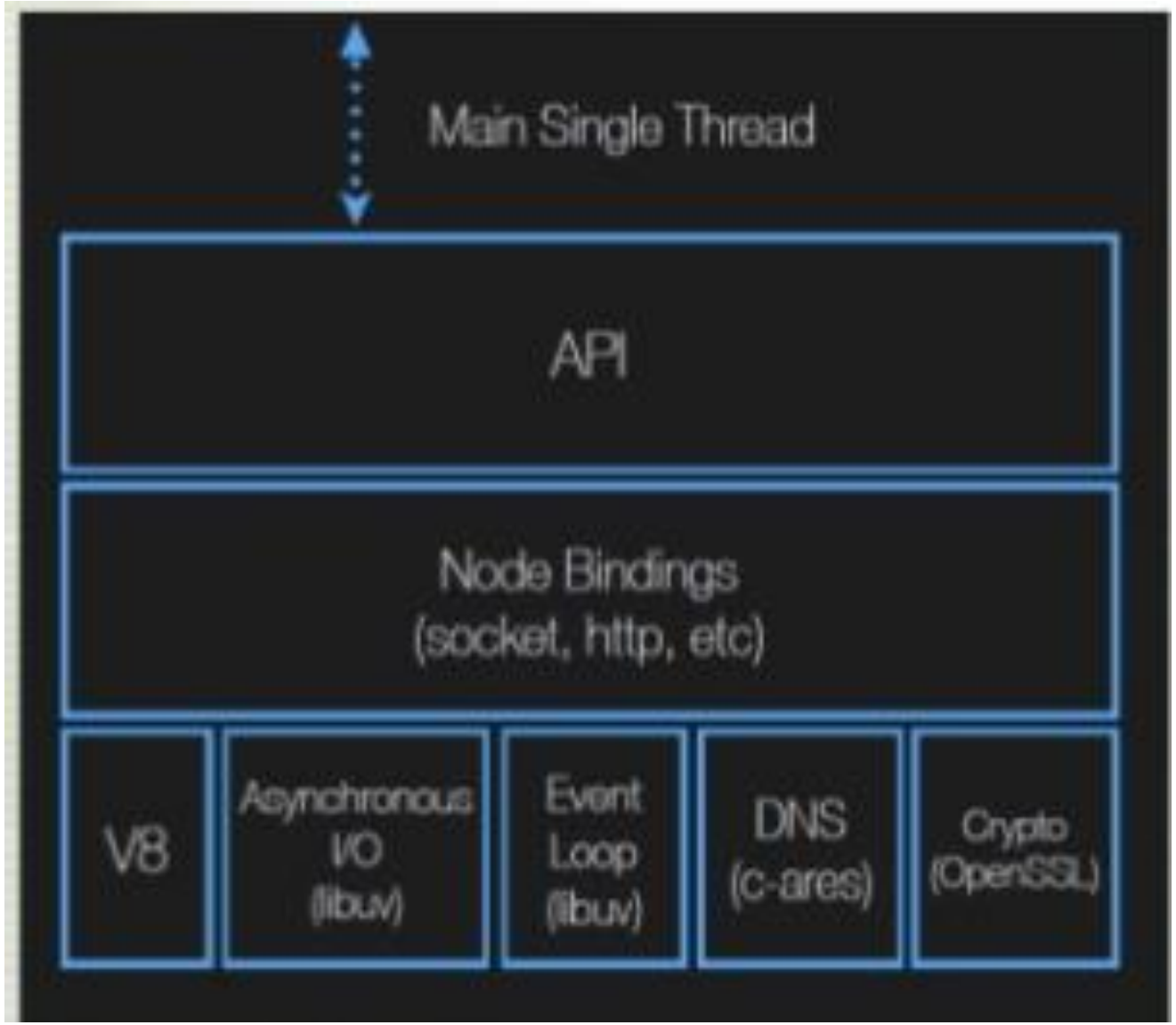
- Node.js possui uma arquitetura orientada a eventos capaz de operações de I/O **assíncronas**.
- Esta escolha de design tem como objetivo otimizar a vazão e escala de requisições em aplicações web com muitas operações de entrada e saída (request e response, por exemplo), bem como aplicações web real-time(como mensageria e jogos).
- O Node.js (ambiente de execução Javascript no servidor) foi implementado baseado no interpretador V8 JavaScript Engine (interpretador de JavaScript em C++ com código aberto do Google, utilizado no Chrome), com desenvolvimento mantido pela fundação Node.js em parceria com a Linux Foundation.

Síncrono x Assíncrono

Modelo NodeJS x Modelo Tradicional



Arquitetura NodeJS



Características NodeJS - Assíncrona

- Node.js é uma tecnologia **assíncrona** que trabalha em uma única thread de execução. Cada requisição ao Node.js não bloqueia o processo do mesmo, atendendo a um volume massivo de requisições ao mesmo tempo, mesmo sendo single thread.
- Imagine que exista apenas um fluxo de execução. Quando chega uma requisição, ela entra nesse fluxo. A máquina virtual JavaScript verifica o que tem de ser feito, delega a atividade (consultar dados no banco, por exemplo) e volta a atender novas requisições enquanto este processamento paralelo está acontecendo. Quando a atividade termina (já temos os dados retornados pelo banco), ela volta ao fluxo principal para ser devolvida ao requisitante.

Características NodeJS - Flexibilidade

- O Node.js possui o gerenciador de pacotes reusáveis **NPM** (Node Package Manager), o maior repositório de softwares, dando ao interpretador um potencial a ser utilizada em qualquer situação. O pacote mais conhecido é o Express.js, um framework completo para desenvolvimento de aplicações Web.
- Um ambiente Node.js não exige muitos recursos computacionais tradicionais. Se utilizado em conjunto com a ferramentas Docker, o ganho na velocidade de deploy e replicação de máquinas pode ser muito significativo e, em ambientes escaláveis, de micro-serviços e serverless, representa menos custo e mais eficiência

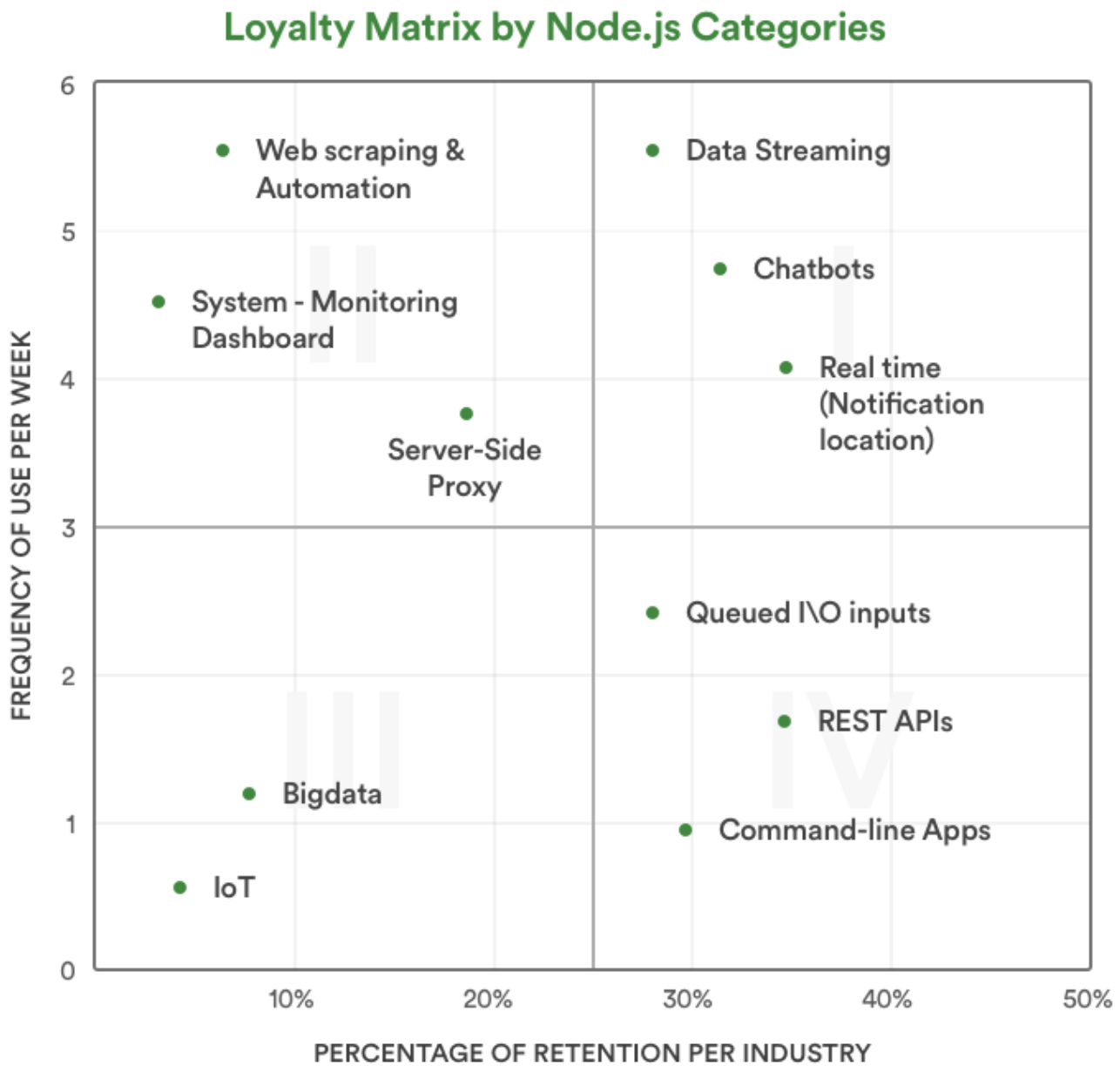
Características NodeJS - Produtividade

- **Vasto repositório de pacotes:** O NPM fornece pacotes de código reusáveis e provavelmente aquela integração que você precisa fazer com outro sistema ou banco de dados já está implementado e disponível gratuitamente para instalar via NPM.
- **Mesma linguagem no frontend e backend:** Javascript é a linguagem padrão para desenvolvimento web client-side. Empresas de desenvolvimento Web contam como esse know-how como um ponto de partida importante para iniciar o uso do Node.js. Além disso, esse fator pode representar ganhos de reutilização de código e criação de equipes multidisciplinares, com melhor aproveitamento de recursos.

Casos de uso

- **Desenvolvimento de APIs:** Um dos principais uso da tecnologia, uma vez que, por padrão, ela apenas sabe processar requisições. Não apenas por essa limitação, mas também porque seu modelo não bloqueante de tratar as requisições o torna excelente para essa tarefa, consumindo pouquíssimo hardware.
- **Aplicações em tempo real:** Usando algumas extensões de web socket com Socket.io, Comet.io, etc é possível criar aplicações de tempo real facilmente sem onerar demais o seu servidor como acontecia antigamente com Java RMI, Microsoft WCF, etc

Casos de uso

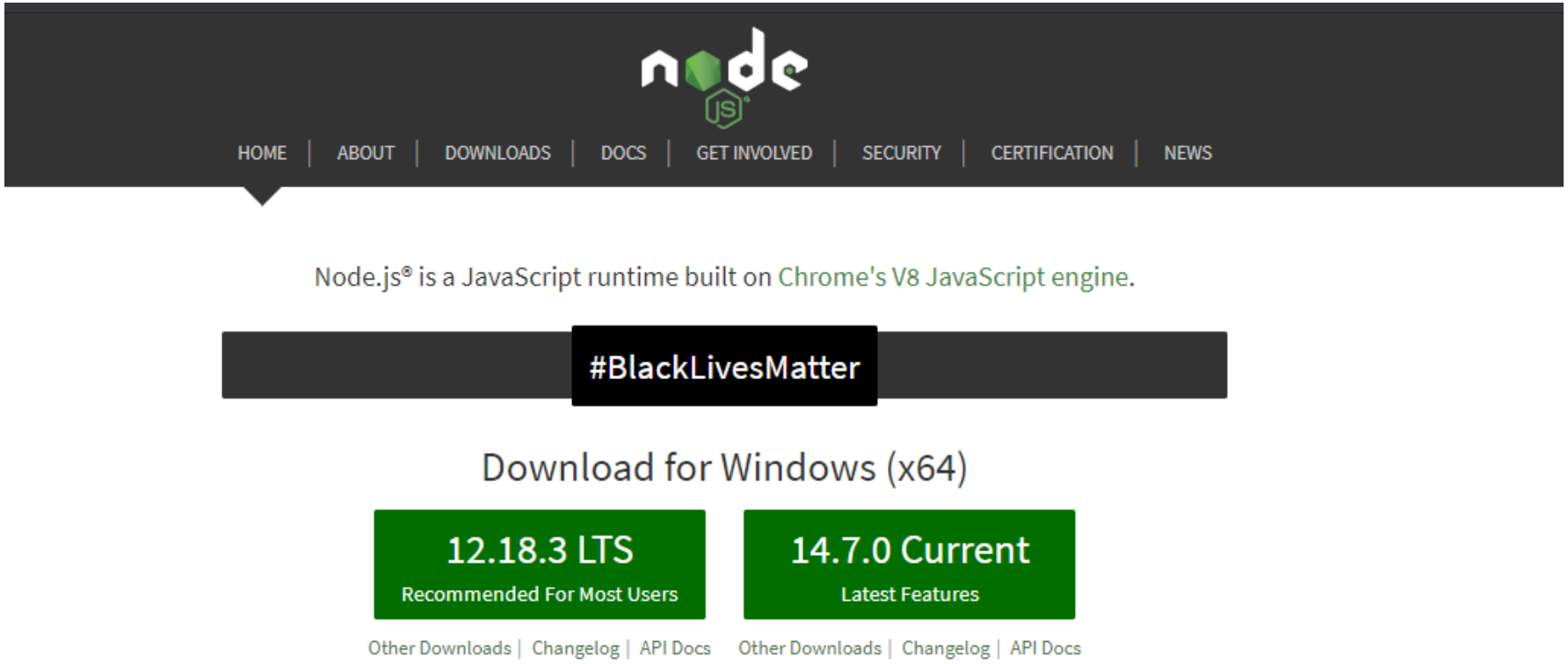

Loyalty Matrix by Node.js Categories
FREQUENCY OF USE PER WEEK
PERCENTAGE OF RETENTION PER INDUSTRY
0
1
2
3
4
5
6
10%
20%
30%
40%
50%
Web scraping & Automation
Data Streaming
System - Monitoring Dashboard
Chatbots
Server-Side Proxy
Real time (Notification location)
Queued I\O inputs
REST APIs
Bigdata
Command-line Apps
IoT

Configurações NodeJS

- Para que seja possível programar com a plataforma Node.js é necessária a instalação da plataforma Node.js. A plataforma é é distribuída gratuitamente pelo seu mantenedor, Node.js Foundation, para diversos sistemas operacionais em seu website oficial.

O Node.js é composto basicamente por:

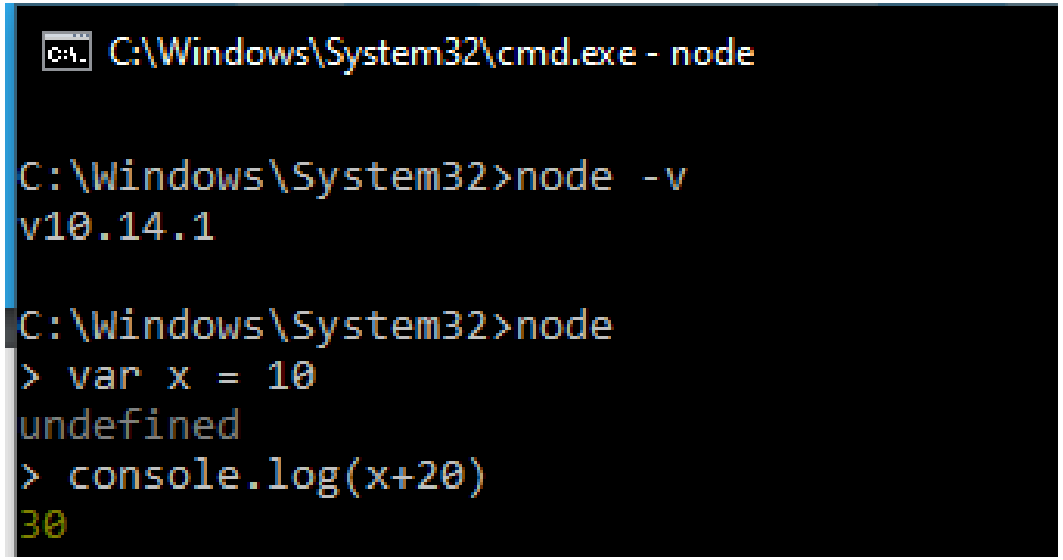
- Um interpretador JavaScript (Google V8, o mesmo do Chrome);
- Uma biblioteca para I/O de baixo nível (libuv);
- Bibliotecas de desenvolvimento básico (os core modules);
- Um gerenciador de pacotes via linha de comando (NPM);
- Um gerenciador de versões via linha de comando (NVM);
- Utilitário REPL via linha de comando;

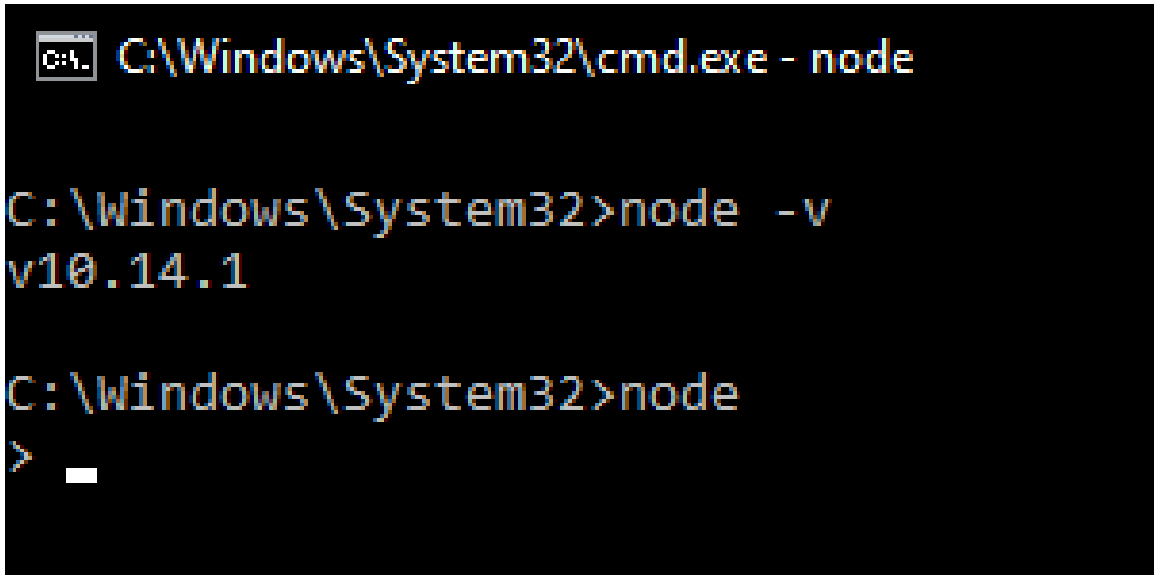


Configurações NodeJS

- Após a instalação do Node, para verificar se ele está funcionando, abra seu terminal de linha de comando (DOS, Terminal, Shell, bash, etc) e digite o comando abaixo:

Ao executar apenas o comando node, será invocado o utilitário **REPL** do Node.js (Read-Eval-Print-Loop) permitindo programação e execução via terminal, linha-a-linha.



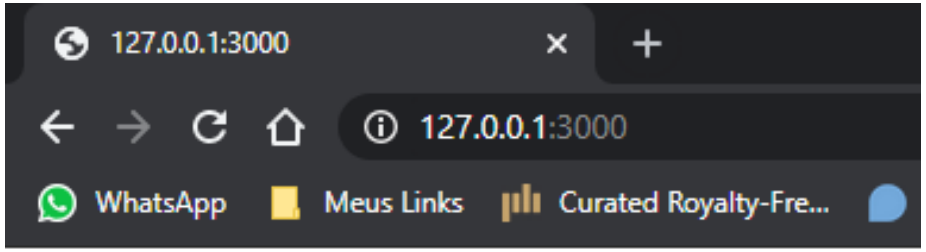


Exemplo NodeJS

server.js

```
1  const http = require('http');
2
3  const hostname = '127.0.0.1';
4  const port = 3000;
5
6  const server = http.createServer((req, res) => {
7    res.statusCode = 200;
8    res.setHeader('Content-Type', 'text/plain');
9    res.end('Hello, World!\n');
10 });
11
12 server.listen(port, hostname, () => {
13   console.log(`Server running at http://${hostname}:${port}/`);
14 });
```

```
C:\Windows\System32\cmd.exe - node server.js

C:\Gravações\2020\Demanda PTI\NodeJS>node server.js
Server running at http://127.0.0.1:3000/
```



Módulos

- No sistema de módulo Node.js, cada arquivo é tratado como um módulo separado.
- Para exportar um módulo usamos a palavra **export** e para importar, usamos o **require**.

square.js

```
1  module.exports = class Square {
2    constructor(width) {
3      this.width = width;
4    }
5
6    area() {
7      return this.width ** 2;
8    }
9  };
```

bar.js

```
1  const Square = require('./square.js');
2  const mySquare = new Square(2);
3  console.log(`The area of mySquare is ${mySquare.area()}`);
```

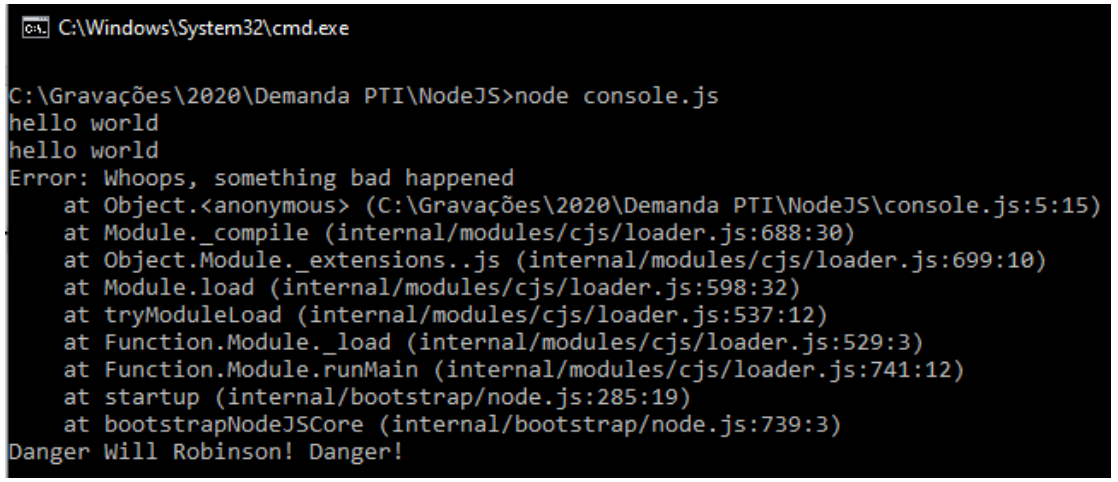
```
C:\Windows\System32\cmd.exe

C:\Gravações\2020\Demanda PTI\NodeJS>node bar.js
The area of mySquare is 4
```

Console

- O módulo do console fornece um console de depuração simples que é semelhante ao mecanismo do console JavaScript fornecido pelos navegadores da web.

console.js

```
1  console.log('hello world');
2  // Prints: hello world, to stdout
3  console.log('hello %s', 'world');
4  // Prints: hello world, to stdout
5  console.error(new Error('Whoops, something bad happened'));
6  // Prints: [Error: Whoops, something bad happened], to stderr
7
8  const name = 'Will Robinson';
9  console.warn(`Danger ${name}! Danger!`);
```

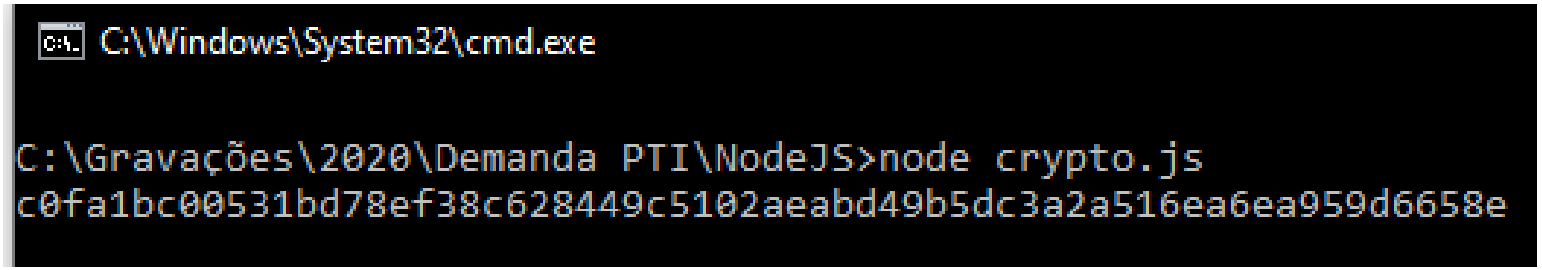


Crypto

- O módulo de criptografia fornece funcionalidade criptográfica que inclui um conjunto de wrappers para funções de hash, HMAC, cifra, decifração, assinatura e verificação do OpenSSL.

crypto.js

```
1  const crypto = require('crypto');
2
3  const secret = 'abcdefg';
4  const hash = crypto.createHmac('sha256', secret)
5                     .update('I love cupcakes')
6                     .digest('hex');
7  console.log(hash);
```



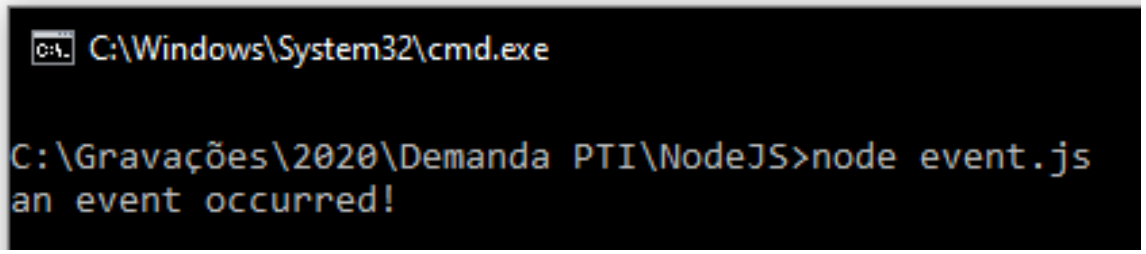
Eventos

- Grande parte da API principal do Node.js é construída em torno de uma arquitetura assíncrona orientada a eventos, na qual certos tipos de objetos (chamados de "emissores") emitem eventos nomeados que fazem com que objetos de função ("ouvintes") sejam chamados.

event.js

```
1  const EventEmitter = require('events');
2
3  class MyEmitter extends EventEmitter {}
4
5  const myEmitter = new MyEmitter();
6  myEmitter.on('event', () => {
7    console.log('an event occurred!');
8  });
9  myEmitter.emit('event');
```

O método eventEmitter.on () é usado para registrar ouvintes, enquanto o método eventEmitter.emit () é usado para disparar o evento.



EventEmitter.On x Once

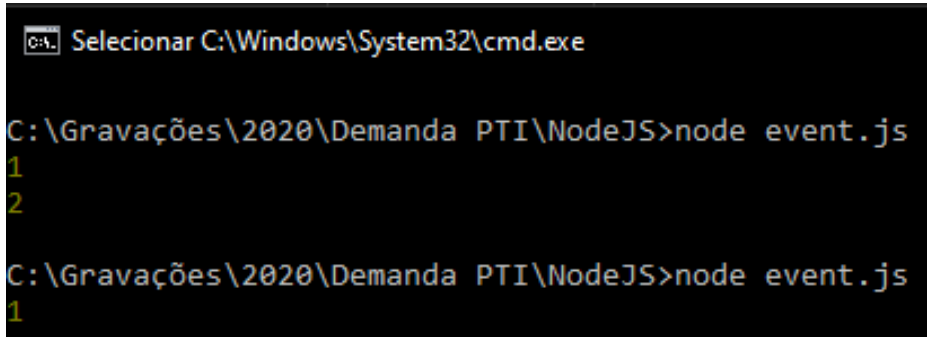
- No método on(), esse ouvinte será chamado toda vez que o evento nomeado for emitido. Já no método once(), o ouvinte que é chamado no máximo uma vez para um determinado evento.

event.js

```
1  const EventEmitter = require('events');
2  class MyEmitter extends EventEmitter {}
3  const myEmitter = new MyEmitter();
4  let m = 0;
5  myEmitter.on('event', () => {
6    console.log(++m);
7  });
8  myEmitter.emit('event');
9
10 myEmitter.emit('event');
```

event.js

```
1  const EventEmitter = require('events');
2  class MyEmitter extends EventEmitter {}
3  const myEmitter = new MyEmitter();
4  let m = 0;
5  myEmitter.once('event', () => {
6    console.log(++m);
7  });
8  myEmitter.emit('event');
9
10 myEmitter.emit('event');
```



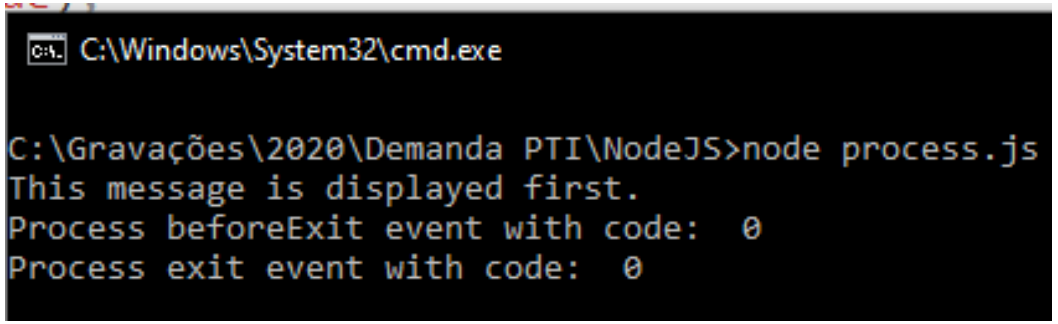
Process

- O objeto de processo é um global que fornece informações e controle sobre o processo atual do Node.js. Como um global, está sempre disponível para aplicativos Node.js sem usar require (). Ele também pode ser acessado explicitamente usando require ().

process.js

```
1  process.on('beforeExit', (code) => {
2    console.log('Process beforeExit event with code: ', code);
3  });
4
5  process.on('exit', (code) => {
6    console.log('Process exit event with code: ', code);
7  });
8
9  console.log('This message is displayed first.');
```

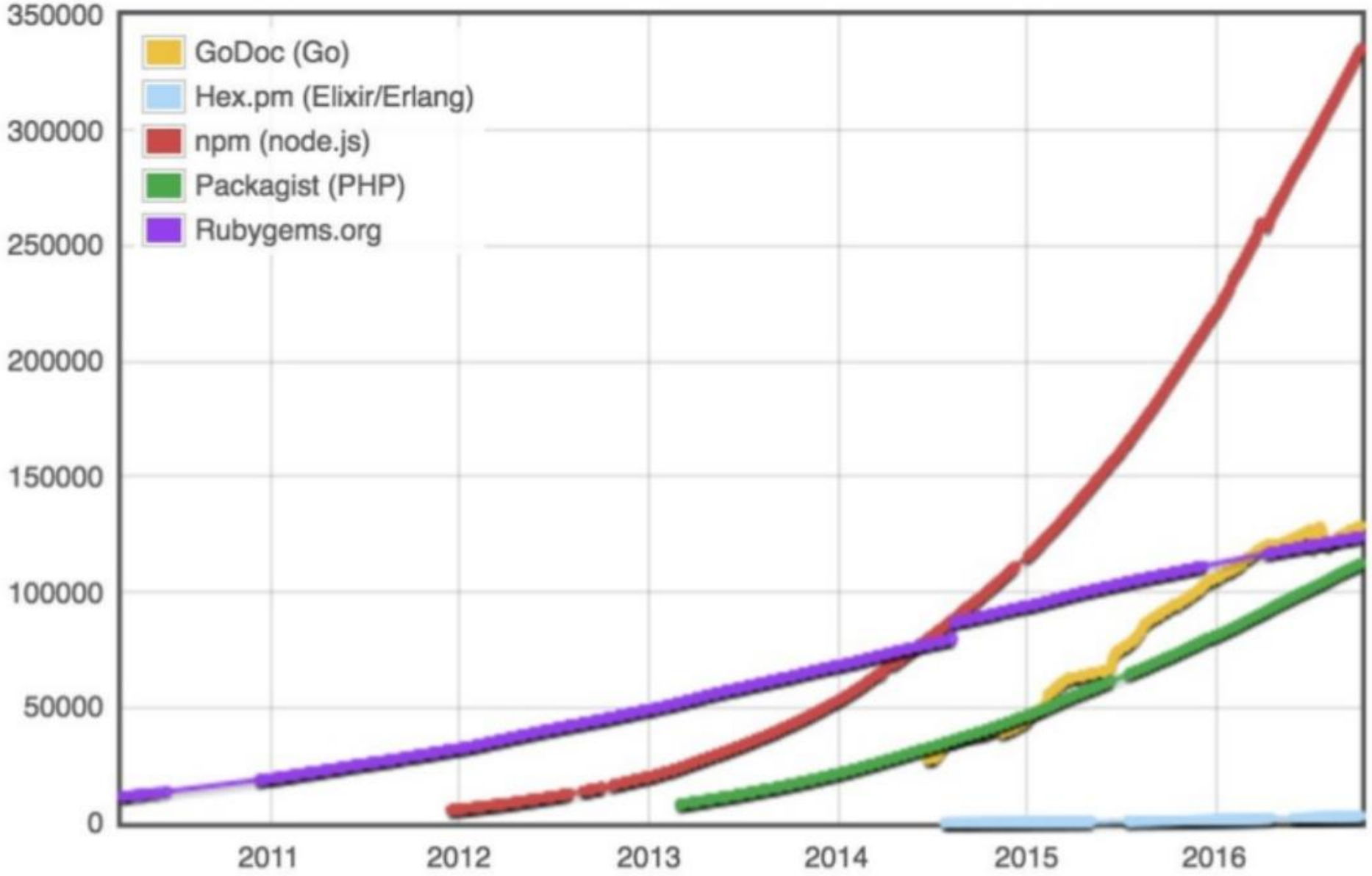
O evento 'beforeExit' é emitido quando o Node.js esvazia seu loop de eventos e não tem trabalho adicional para agendar.



NPM


npm is the package manager for javascript.
351,762
total packages
84,313,758
downloads in the last day
1,353,327,940
downloads in the last week
5,739,865,435
downloads in the last month
https://www.npmjs.com

NPM



https://www.npmjs.com

NPM

- Para inicializar um projeto com o npm, vá ao terminal e execute o comando **npm init**.
- Caso já tenha um arquivo package.json e quiser baixar as bibliotecas javascript do projeto, é só executar o comando **npm install**.

package.json

```
{
  "name": "teste",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Rodrigo Macedo <rcbm539@gmail.com>",
  "license": "MIT",
  "dependencies": {
    "axios": "^0.19.2"
  }
}
```

Você pode baixar diversas bibliotecas para o seu projeto. Simplesmente execute npm install + [nome da biblioteca]. Caso queira adicionar a biblioteca dentro do arquivo.json, acrescente o parâmetro –save ao comando.

```
C:\Gravações\2020\Demanda PTI\NodeJS\teste>npm i axios --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN teste@1.0.0 No description
npm WARN teste@1.0.0 No repository field.

+ axios@0.19.2
added 4 packages from 7 contributors and audited 4 packages in 10.385s
found 0 vulnerabilities
```

Q1) [FGV MPE AL 2018] No script NodeJS

```
exports.myDateTime = function () {
    return Date();
};
```

o papel do termo exports é

a) definir o nome da biblioteca corrente como “exports”.

b) definir um método para a classe exports.

c) definir uma função externa que ficará localizada no módulo exports.

d) fazer referência a uma variável externa localizada no módulo exports.

e) permitir que a função myDataTime seja acessível para outros módulos.

Q1) [FGV MPE AL 2018] No script NodeJS

```
exports.myDateTime = function () {
    return Date();
};
```

o papel do termo exports é

a) definir o nome da biblioteca corrente como “exports”.

b) definir um método para a classe exports.

c) definir uma função externa que ficará localizada no módulo exports.

d) fazer referência a uma variável externa localizada no módulo exports.

e) permitir que a função myDataTime seja acessível para outros módulos.

Q2) [FGV MPE AL 2018] Observe a linha de código NodeJS exibida a seguir.

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

Assinale a opção que melhor descreve esse comando.

a) Criar um objeto da classe require.

b) Criar um objeto HTMLrequest.

c) Disparar um request HTML.

d) Incluir o módulo HTTP.

e) Invocar a função require, definida pelo usuário.

Q2) [FGV MPE AL 2018] Observe a linha de código NodeJS exibida a seguir.

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

Assinale a opção que melhor descreve esse comando.

a) Criar um objeto da classe require.

b) Criar um objeto HTMLrequest.

c) Disparar um request HTML.

d) Incluir o módulo HTTP.

e) Invocar a função require, definida pelo usuário.

Q3) [FGV AL RO 2018] No contexto do Node.js. analise a linha de código a seguir.

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

Assinale a opção que apresenta a correta interpretação dessa função.

a) Abrir uma nova conexão.

b) Estabelecer o nível de segurança de uma conexão.

c) Ler uma página do endereço default.

d) Permitir a utilização do protocolo http.

e) Promover a inclusão do módulo http.

Q3) [FGV AL RO 2018] No contexto do Node.js. analise a linha de código a seguir.

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

Assinale a opção que apresenta a correta interpretação dessa função.

a) Abrir uma nova conexão.

b) Estabelecer o nível de segurança de uma conexão.

c) Ler uma página do endereço default.

d) Permitir a utilização do protocolo http.

e) Promover a inclusão do módulo http.

Q4) [IFSP IFSP 2019] No início da popularização da Web e até há alguns anos atrás, algumas das alternativas, mais comuns para a inserção de conteúdos dinâmicos e de multimídia em páginas Web era a utilização de controles ActiveX, tecnologia da Microsoft, a utilização de Applets Java, tecnologia atualmente da Oracle e a utilização de animações em Flash, que é um aplicativo de editoração da Adobe, que foi renomeado para Adobe Animate no ano de 2018. Hoje em dia, a utilização dessas tecnologias é desencorajada ou não está mais disponível nos navegadores modernos. A alternativa atual para a utilização de tais tipos de conteúdo é viabilizada pelo HTML5 e pelo JavaScript.

Dada a popularidade do JavaScript e a velocidade com que algumas engines, como a V8 do Google, conseguem executar código JavaScript, assinale a alternativa que contém o nome do ambiente de execução ou a tecnologia que permite a utilização do JavaScript do lado do servidor.

a) JavaServer Pages.

b) Apache Tomcat.

c) Node.js.

d) JavaServer Faces.

Q4) [IFSP IFSP 2019] No início da popularização da Web e até há alguns anos atrás, algumas das alternativas, mais comuns para a inserção de conteúdos dinâmicos e de multimídia em páginas Web era a utilização de controles ActiveX, tecnologia da Microsoft, a utilização de Applets Java, tecnologia atualmente da Oracle e a utilização de animações em Flash, que é um aplicativo de editoração da Adobe, que foi renomeado para Adobe Animate no ano de 2018. Hoje em dia, a utilização dessas tecnologias é desencorajada ou não está mais disponível nos navegadores modernos. A alternativa atual para a utilização de tais tipos de conteúdo é viabilizada pelo HTML5 e pelo JavaScript.

Dada a popularidade do JavaScript e a velocidade com que algumas engines, como a V8 do Google, conseguem executar código JavaScript, assinale a alternativa que contém o nome do ambiente de execução ou a tecnologia que permite a utilização do JavaScript do lado do servidor.

a) JavaServer Pages.

b) Apache Tomcat.

c) Node.js.

d) JavaServer Faces.

Q5) [FGV IBGE 2016] A manipulação de eventos assíncronos é preponderante no desenvolvimento de aplicações no Node.js. Os métodos comuns à classe EventEmitter são:

a) .addListerner e .skipListerner;

b) .dropListerner e .dropAllListerner;

c) .onListerner e .emit;

d) .even e .off;

e) .on e .once.

Q6) [CESPE SLU-DF 2019] Julgue o próximo item, relativos à linguagem de programação JavaScript e às ferramentas Node e React.

O Node.js é capaz de gerar conteúdos dinâmicos rodando JavaScript no servidor, porém não tem a capacidade de acessar banco de dados.

Q5) [FGV IBGE 2016] A manipulação de eventos assíncronos é preponderante no desenvolvimento de aplicações no Node.js. Os métodos comuns à classe EventEmitter são:

a) .addListerner e .skipListerner;

b) .dropListerner e .dropAllListerner;

c) .onListerner e .emit;

d) .even e .off;

e) .on e .once.

Q6) [CESPE SLU-DF 2019] Julgue o próximo item, relativos à linguagem de programação JavaScript e às ferramentas Node e React.

O Node.js é capaz de gerar conteúdos dinâmicos rodando JavaScript no servidor, porém não tem a capacidade de acessar banco de dados. ERRADO.

Q7) [FGV IBGE 2017] Analise o trecho de código a seguir escrito em NodeJS:

```
const EventEmitter = require('events');

const emitter = new EventEmitter();
let m = 0;
emitter.once('event', (data) => {
  console.log(++m +":"+data);
});

emitter.emit('event', m);
emitter.emit('event', m);
```

O que será mostrado no console JavaScript do navegador após a execução desse trecho de código é:

a) 1:0

b) 2:1

c) 1:1

d) 1:1
   2:2

e) 1:0
   2:1

Q7) [FGV IBGE 2017] Analise o trecho de código a seguir escrito em NodeJS:

```
const EventEmitter = require('events');

const emitter = new EventEmitter();
let m = 0;
emitter.once('event', (data) => {
  console.log(++m +":"+data);
});

emitter.emit('event', m);
emitter.emit('event', m);
```

O que será mostrado no console JavaScript do navegador após a execução desse trecho de código é:

a) 1:0

b) 2:1

c) 1:1

d) 1:1
   2:2

e) 1:0
   2:1

GABARITO

Q1 – LETRA E.

Q2 - LETRA D.

Q3 – LETRA E.

Q4 - LETRA C.

Q5 - LETRA E.

Q6 - ERRADO.

Q7 – LETRA A.