Primeiros Passos com Node.js

INSTRUTOR: VITOR BRUNO DE OLIVEIRA BARTH

DISCIPLINA: LINGUAGENS DE PROGRAMAÇÃO III

PROFESSOR: JOÃO PAULO PRETI

Instituto Federal de Educação, Ciência e Tecnologia de Mato Grosso

Cuiabá, 06 de Julho de 2018

Agenda

- Apresentação do Ambiente Node.js
- “Olá Mundo” com Node.js
- Comentários sobre *JavaScript*
- Primeiro Servidor Web com Node.js
- Primeira Aplicação com Node.js

Apresentação do Ambiente Node.js

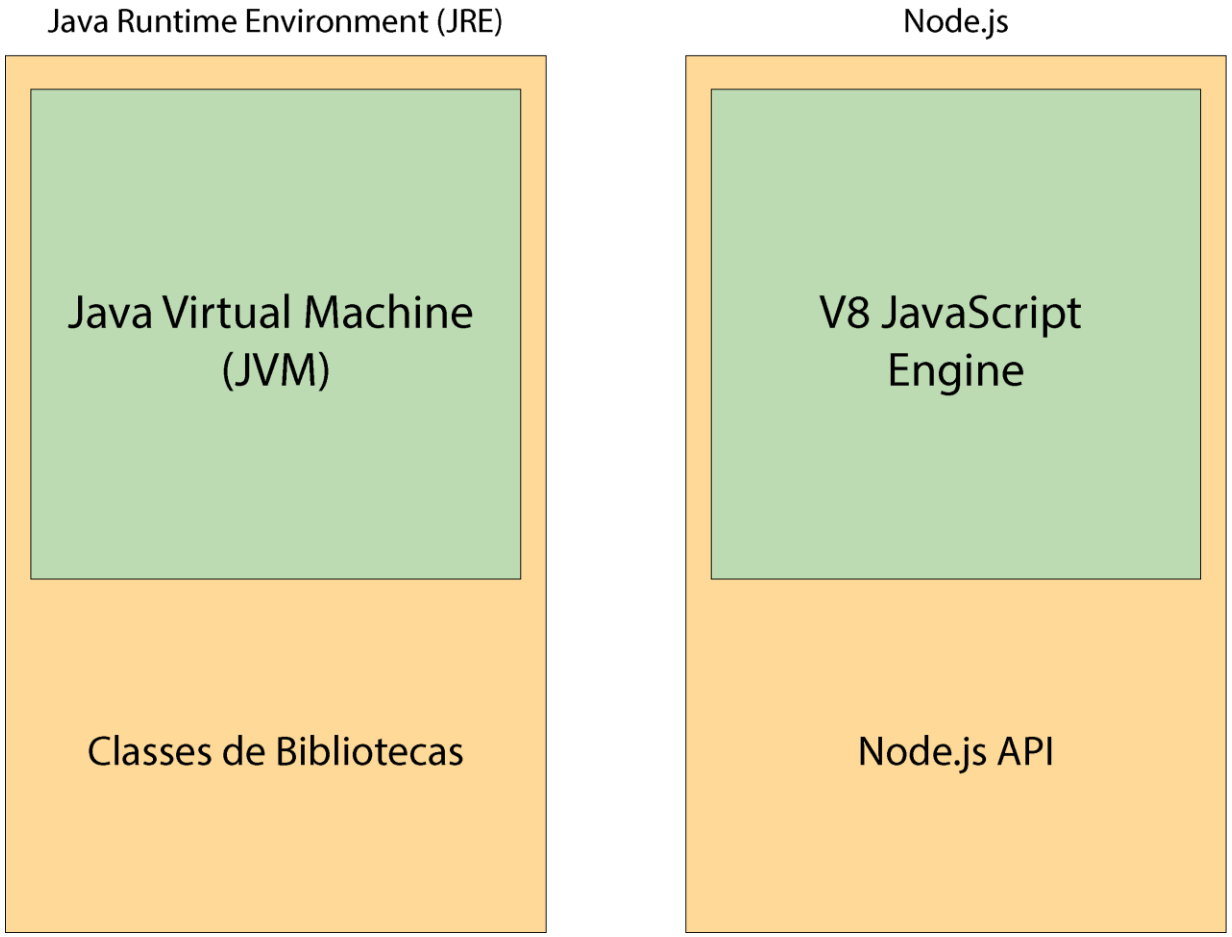

Figura 01 – Comparação com Java

- *Runtime* para *JavaScript*
- Construído sobre o Interpretador *JavaScript* V8
- Modelo Orientado a Eventos
- E/S Não Bloqueante

Apresentação do Ambiente Node.js

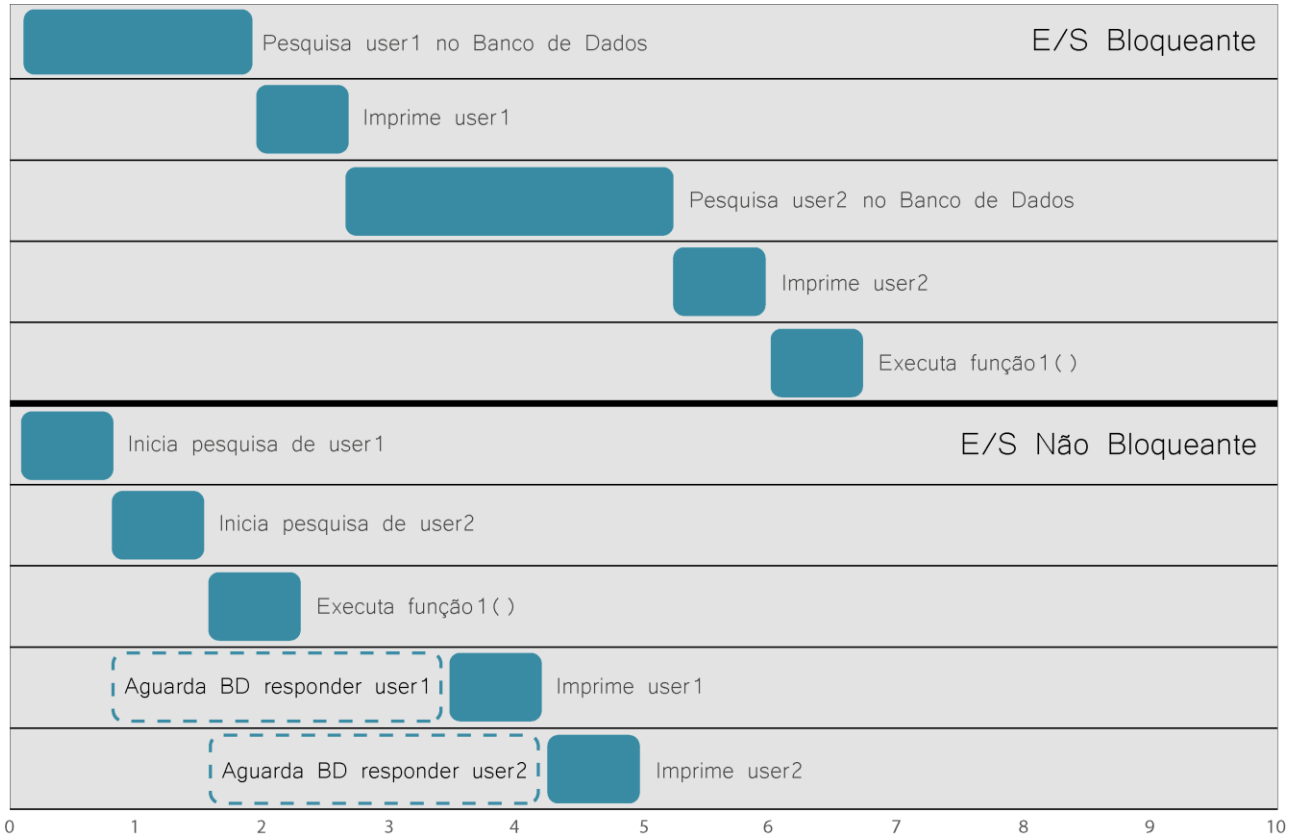

Figura 02 – E/S Bloqueante vs. Não-Bloqueante

- *Runtime* para *JavaScript*
- Construído sobre o Interpretador *JavaScript* V8
- Modelo Orientado a Eventos
- **E/S Não Bloqueante**

Olá Mundo!

```
console.log(“Olá Mundo!”)
```

- (Sim, é só isso!)
- Salve com a extensão **.js**
- Execute via Terminal/PowerShell/Console com o comando:

```
$ node arquivo.js
```

Alguns comentários sobre JS

- É uma dos três pilares da WWW, junto com HTML e CSS
- Multiparadigma: Orientada a Eventos, Imperativa e Funcional
- Interpretada
- Fracamente Tipada

Alguns comentários sobre JS

- Declaração de variáveis pode ser feita:
  - Diretamente: `x = 2`
  - Através das palavrar reservadas *let* ou *var*:

```
let x = 2, y = 2
var soma = x + y
```

- Loops: *for, do, do-while*
- Comparativos: *if, else if, else*
- Comparadores: >, <, <=, >=, ==, ===, !=, &&, ||

Alguns comentários sobre JS

- Declaração de funções pode ser feita:
  - Explicitamente: `function funcao(a, b) { }`
  - *Lambda:* `let funcao = (a, b) => { }`
- Comentários: `//` (linha) ou `/* */` (bloco)
- Ponto e Vírgula ao final da linha é opcional

Primeiro Servidor Web

- Instale o módulo *express*:

```
$ npm install express
```

Primeiro Servidor Web

```
const express = require(‘express’)
const app = express()
const port = 3000

app.get('/', (request, response) => {
  response.send(“Meu primeiro Servidor Web”)
})

app.listen(port, (err) => {
  if (err) return console.log(“Ops... Algo deu errado: ” + err)
  else console.log(“Acesse http://localhost:” + port)
})
```

Acessando Parâmetros

```
[...]

app.get(‘/numero/:numero’, (request, response) => {
  response.send(request.params.numero)
})

[...]
```

Servindo um Arquivo

```
[...]

app.get(‘/', (request, response) => {
  response.sendFile(__dirname + “/index.html”)
})

[...]
```

Servindo Arquivos

```
[...]

app.get('/', (request, response) => {
  response.sendFile(__dirname + “/index.html”)
})

app.get(‘/*', (request, response) => {
  response.sendFile(__dirname + `/${request.params[0]}` )
})


[...]
```

Usando Sockets

- Instale o módulo *socket.io*:

```
$ npm install socket.io
```

Primeiro App com Sockets

```
const express = require('express')
const app = express()
const server = require('http').createServer(app)
const io = require('socket.io')(server)
const port = 3000

server.listen(port, (err) => {
  if (err) return console.log("Ops... Algo deu errado: " + err)
  else console.log("Acesse http://localhost:" + port)
})

app.use(express.static(__dirname + '/public'))
```

/projeto/index.js

Primeiro App com Sockets

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
    <meta charset="UTF-8">
    <title>Primeiro App com Sockets</title>
    <script src="https://code.jquery.com/jquery-1.10.2.min.js"></script>
    <script src="/socket.io/socket.io.js"></script>
    <script src="/index.js"></script>
</head>

<body>
    <h1>Meu Primeiro App</h1>
</body>

</html>
```

/projeto/public/index.html

Primeiro App com Sockets

```
var socket = io.connect('http://localhost:3000’);
```

/projeto/public/index.js

```
io.on('connection', function (socket) {
    console.log('Um usuário se conectou')
});
```

/projeto/index.js