Universidade Estadual do Ceará - UECE
Centro de Ciência e Tecnologia - CCT
Licenciatura em Computação
Disciplina Fundamentos de Engenharia de Software
Semestre 2020.1
Prof. Me. Marcos Eduardo da Silva Santos

Introdução ao Node.js

Thiago Dias de Carvalho Quaresma Gama

Índice

Definição do Node.js

- Plataforma construída sobre o motor JavaScript V8 do Google Chrome para construir aplicações de rede rápidas e escaláveis. Site oficial: https://nodejs.org/
- Utiliza o modelo de Programação Orientada a Eventos, como no JavaScript *client-side*.
- Adota um padrão não obstrutivo de escrever código, ou seja, você consegue estruturar seu código de maneira que operações não dependam de nada que está sendo executado para serem executadas de forma independente.

Callbacks em Node.js

- São funções que serão executadas de modo assíncrono, ou posteriormente.
- A chave para a compreensão de *callbacks* é perceber que eles são utilizados sempre que você não sabe quando alguma operação assíncrona estará completa, mas você sabe quando se completará.
  - A última linha da função assíncrona.

Atividades Práticas

A seguir serão apresentadas uma lista de atividades práticas complementares ao exemplo mostrado no artigo

Atividade 1

- Faça a instalação do Node.js (https://nodejs.org/)
- Acesse o site NodeSchool (https://nodeschool.io/)
- Clique na opção Tutorials (https://nodeschool.io/#workshoppers)
- Abra o Terminal (Prompt de Comando, PowerShell, etc.)
- Instale os tutoriais do NodeSchool:
  - `$ npm install -g learnyounode`

Atividade 1

- Navegue até a pasta onde deseja criar os códigos dos exercícios:
  - `$ cd \Users\thiag\OneDrive\Área de Trabalho\atividade_1`
- Abra os exercícios do NodeSchool:
  - `$ learnyounode`
- Selecione o exercício desejado, por exemplo:
  - `HELLO WORLD`
- Analise os requisitos de cada exercício.
- A seguir serão exibidas as instruções para os 4 primeiros.

Atividade 1

- Abra um editor de texto, como o Visual Studio Code (https://code.visualstudio.com/), e navegue até a passados exercícios.
- Crie o arquivo para o exercício 1, com a extensão .js, por exemplo, **exercicio1.js**, e inicie a escrita dos código JavaScript para a realização do o exercício.
- Para testar a execução de exercicio1.js, rode:
  - `$ learnyounode run exercicio1.js`
- Ao concluir o exercício, valide seu código:
  - `$ learnyounode verify exercicio1.js`

Atividade 1

- Caso o exercício esteja correto, o learnyounode conterá uma mensagem informando o acerto e a solução oficial do exercício:
- **✓ Submission results match expected**
- **# PASS Your solution to HELLO WORLD passed!**
- Caso o exercício esteja incorreto, a resposta conterá os detalhes do erro, o local onde foi encontrado e o texto:
- **✗ Submission results did not match expected!**
- **# FAIL Your solution to HELLO WORLD didn't pass. Try again!**

Atividade 2

- Escreva um programa que aceite um ou mais números como argumentos de linhas de comando e imprima a soma destes números no console.
- Argumentos de linha de comando são acessíveis pelo objeto global **process**.
- O objeto **process** tem uma propriedade chamada **argv** que é um vetor que contém a linha de comando completa: **process.argv**.
- Executando **`$ node atividade2.js 1 2 3`**, retornará o vetor:
- **`['node', '/pasta/da/atividade2.js', '1', '2', '3']`**

Atividade 3

- Escreva um programa que use apenas uma operação síncrona que leia um arquivo e imprima no console o número de novas linhas que ele contém.
- Será necessário utilizar o módulo **fs**, da biblioteca central do Node.js. Para carregar este tipo de módulo, use a seguinte sintaxe:
- `var fs = require('fs');`
- O caminho completo do arquivo será fornecido como o primeiro argumento de linha de comando **`(process.argv[2])`**.

Atividade 4

- Escreva um programa que use apenas uma operação assíncrona que leia um arquivo e imprima no console o número de novas linhas que ele contém.
- O caminho completo do arquivo será fornecido como o primeiro argumento de linha de comando.
- A solução para este problema é quase a mesma para o problema anterior, porém você terá de fazer agora no modo Node.js assíncrono.

Atividade 4

- Ao invés de **fs.readFileSync()** você vai usar **fs.readFile()** e ao invés de retornar um valor deste método você precisa coletar o valor a partir de uma função *callback* que você irá passar como segundo parâmetro.
- Funções *callback* no Node.js normalmente possuem a assinatura abaixo:
- `function callback(err, dados) { /* ... */ }`
- Com isso você pode checar se ocorreu um erro por verificar se o primeiro parâmetro é válido. Caso não exista erro, você poderá ter seu objeto **Buffer** como segundo parâmetro.

Referências

- Künzel, P. (2019) “Introdução ao Node.js”, https://www.mundojs.com.br/2019/05/31/introducao-ao-node-js/, Abril.
- Eler, E. (2016) “Introdução ao Node.js”, https://pt.slideshare.net/edgareler/introduo-ao-nodejs-62352618, Abril.

Obrigado!

Alguma pergunta?

Você pode me contatar em thiagoddcqg@gmail.com