FIC 2 – Módulo II

Aula 2 - Servidor Web no Node

Criação de páginas customizadas com Node.js

Leonardo Dalvi Pio

INSTITUTO FEDERAL
Espírito Santo

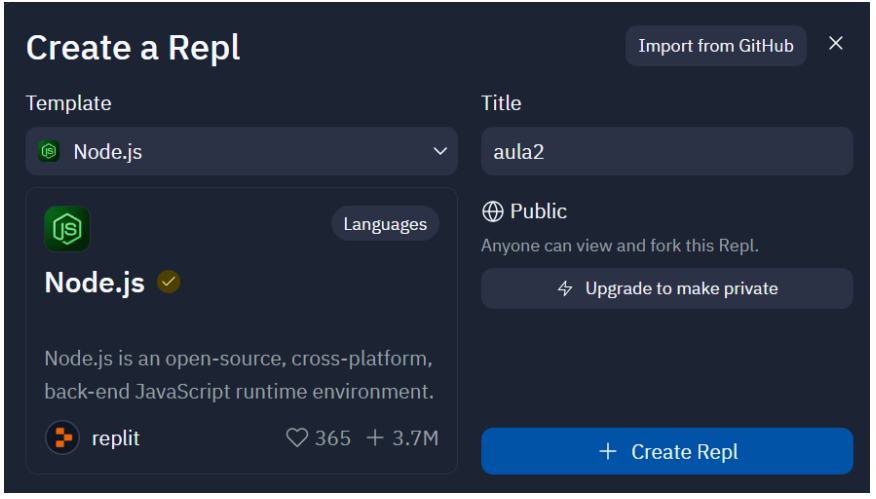
Servidor Web

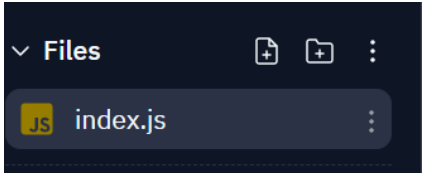
- Criar um Servidor Web em Node.
- Página com códigos de erros (HTTP Cats).
- No Replit, se no pacote/recurso não estiver disponível, ele baixará automaticamente.

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- No Replit, crie uma pasta nomeada "Aula 2".
- Dentro da pasta, crie um novo arquivo (New Repl). Selecione o Template 'Node.js' e atribua o título como: "aula2". Será incluído o arquivo 'index.js' em *Files*.





INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Primeiramente devemos criar um objeto ‘http’ e trabalhar com ele com **requisição** e **resposta**.
- É realizada uma **requisição** e o objeto retorna uma **resposta** para quem solicitou a **requisição**.
- Devemos indicar a porta de comunicação do servidor.

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Criaremos uma constante nomeada ‘http’ conforme código:

```
const http = require('http')
```

- Utilizamos o *require* (requisição) que usaremos para acessar o módulo 'http’ (importação do módulo `http`).
- Utilizaremos a porta 443.

```
const porta = 443
```

- Note que não é necessário utilizar o “;” (ponto-e-vírgula).

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Em seguida, com o objeto 'http', utilizaremos a função createServer().
- Utilizaremos um recurso que se destacou muito no *JavaScript*: as *Arrow Functions*. Passaremos às devidas variáveis: **req (requisição)** e **res (resposta)**, também bastante utilizadas em inglês, sendo *request* e *response*. As *Arrow Functions* são utilizadas por: `=>`.

```
const servidor = http.createServer((req, res) => {
```

- Obrigatoriamente, devemos utilizar nessa ordem, requisição e resposta.

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Na nossa *Arrow Functions* criada, começaremos com a **resposta (res)**. Utilizaremos o writeHead(status:Code, reasonPhrase?), com código 200 (conforme Página http.cat – status **ok**) e cabeçalho 'Content-Type':'text/plain'.

```
  res.writeHead(200, { 'Content-Type': 'text/plain' })
```

- Continuando na **res**, incluiremos o comando 'write', escrevendo o texto `Primeiro Servidor Web`.

```
res.write('Primeiro servidor Node')
```

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Indicamos o final da nossa **resposta** com o comando 'end()'.

```
res.end()
```

- Para finalizar, incluiremos o comando '.listen()' e a porta de comunicação (porta 443) com o servidor, apresentando a mensagem: "Servidor rodando" (com o comando: 'console.log').

```
servidor.listen(porta, () => {console.log('Servidor rodando') })
```

- Pronto! Já temos um Servidor Web!

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

Codificação final:

```
1  const http = require('http')
2  const porta = 443
3  const servidor = http.createServer((req, res) => {
4    res.writeHead(200, { 'Content-Type': 'text/plain' })
5    res.write('Primeiro servidor Node')
6    res.end()
7  })
8  servidor.listen(porta, () => {console.log('Servidor rodando') })
```

INSTITUTO FEDERAL
Espírito Santo

Testes!

- Para testar o nosso Servidor Web, inicie com o botão 'Run' do Replit.
- Na aba 'Console' é possível observar a mensagem: "Servidor rodando".
- Na aba 'Webview' é possível observar escrito: "Primeiro servidor Node".

INSTITUTO FEDERAL
Espírito Santo

Criando um Servidor Web em Node

- Note que não fizemos nenhuma **requisição** (ainda).
- Para incluir a **requisição**, devemos trabalhar com as Interfaces de Programação de Aplicação (*Application Programming Interface - APIs)* das *Rows*, o que será apresentado mais adiante.

INSTITUTO FEDERAL
Espírito Santo

INSTITUTO
FEDERAL
Espírito Santo