

FIC 2 – Módulo II

Aula 5 – Introdução ao Framework Express do Node

Criação de páginas
customizadas com Node.js

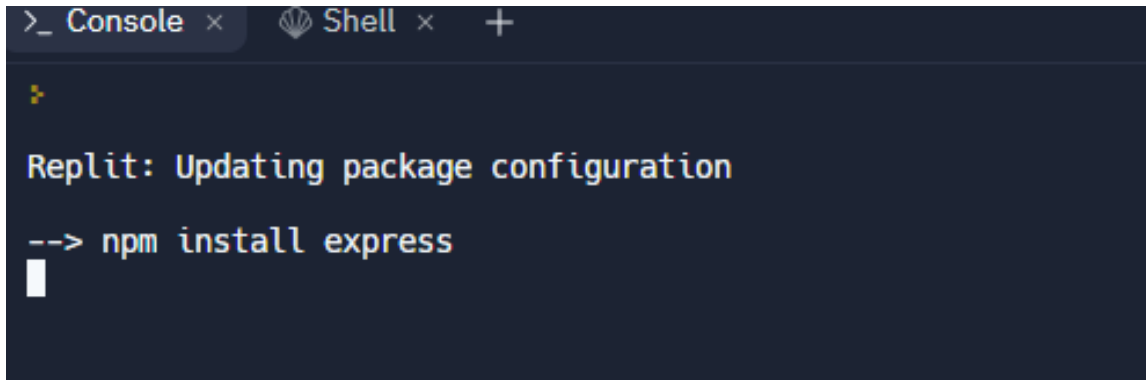
Leonardo Dalvi Pio



INSTITUTO FEDERAL
Espírito Santo

Framework Express do Node

- Apresentar o *Framework Express* do Node.
- No Replit, se no pacote/recurso (***pagkage.json***, ***express***) não estiver disponível, ele baixará automaticamente.



```
>_ Console x Shell x +  
  
Replit: Updating package configuration  
--> npm install express  
█
```

Framework Express do Node

- *Framework Express:*
 - Principal *Framework* do Node.
 - Aplicações web (*Back-end*).
 - Gerenciar requisições (HTTP em diferentes URLs e Portas).
 - Minimalista (comandos e recursos simples).
 - Agilidade na programação.



Framework Express do Node

- Em um site tradicional baseado em dados, um aplicativo da Web aguarda pedidos HTTP do navegador da web (ou outro cliente). Quando um pedido (requisição) é recebido, o aplicativo descreve quais ações são necessárias com base no padrão de URL e possivelmente informações associadas contidas em dados *POST* ou *GET*.

Framework Express do Node

- Dependendo do que é necessário, pode-se ler ou escrever informações em um banco de dados ou executar outras tarefas necessárias para satisfazer a solicitação. O aplicativo retornará uma **resposta** ao navegador da Web, criando, de forma dinâmica, uma página HTML para o navegador, exibindo e inserindo os dados recuperados em espaços reservados em um modelo HTML.

Métodos do Framework Express

- Métodos para especificar qual função é chamada quando chega requisição HTTP (*GET, POST, SET*, etc.).
- Rotas e métodos para especificar o mecanismo de modelo ('view') usado, onde o modelo arquivos estão localizados e qual modelo usar para renderizar uma resposta.

Primeiro exemplo utilizando o Express

```
1  const express = require('express')
2  const app = express()
3  const porta = 443
4
5  ✓ app.get('/', function(req, res) {
6    res.send('Olá, Mundo! Estou usando o Express!')
7  })
8
9  ✓ app.listen(porta, function() {
10    console.log('App rodando!')
11  })
```



Primeiro exemplo utilizando o Express

- As duas primeiras linhas `'require()'` importam o módulo *Express* e criam uma aplicação *Express*. Esse objeto (tradicionalmente nomeado de `app`), tem métodos de roteamento de requisições HTTP, e várias outras configurações e recursos (por exemplo, o modo de ambiente, se as definições de rota são sensíveis a maiúsculas e minúsculas, etc).

```
const express = require('express')  
const app = express()  
const porta = 443
```


Primeiro exemplo utilizando o Express

- A parte do meio do código (as três linhas que começam com `'app.get'`) mostra uma definição de rota. O método `'app.get()'` especifica uma função de retorno de chamada que será invocada sempre que exista uma solicitação HTTP *GET* com um caminho `('/')` relativo à raiz do site. A função de retorno de chamada requer uma solicitação e um objeto de resposta como argumentos, e simplesmente chama `'send()'` na resposta para retornar a *string*: "Olá Mundo! Estou usando o Express!".

```
app.get('/', function(req, res) {  
  res.send('Olá, Mundo! Estou usando o Express!')  
})
```

Primeiro exemplo utilizando o Express

- O bloco final inicia o servidor na porta '443' e imprime um comentário de *log* no console. Com o servidor em execução, você pode visualizar na aba 'Webview' do Replit ou no seu navegador para ver o exemplo de resposta retornado.

```
app.listen(porta, function() {  
  console.log('App rodando!')  
})
```

Manipulação de rotas utilizando o Express

- No nosso “Olá Mundo” em *Express* (slides anteriores), nós definimos uma (*callback*) função manipuladora de rota para requisição *GET HTTP* para a raiz do site ('/').

```
app.get('/', function(req, res) {  
  res.send('Olá, Mundo! Estou usando o Express!')  
})
```

Manipulação de rotas utilizando o Express

- A função de retorno de chamada requer uma solicitação e um objeto de resposta como argumento. Neste caso, o método simplesmente chama `'send()'` na resposta para retornar a string "Olá Mundo! Estou usando o Express!" Há uma série de outros métodos de resposta para encerrar o ciclo de solicitação/resposta, por exemplo, você poderia chamar `'res.json()'` para enviar uma resposta *JSON* ou `'res.sendFile()'` para enviar um arquivo.

Manipulação de rotas utilizando o Express

- O *Express* também fornece métodos para definir manipuladores de rotas para todas as outras requisições *HTTP*, que são usadas exatamente da mesma maneira: *post()*, *put()*, *delete()*, *options()*, *trace()*, *copy()*, *lock()*, *mkcol()*, *move()*, *purge()*, *propfind()*, *proppatch()*, *unlock()*, *report()*, *mkactivity()*, *checkout()*, *merge()*, *m-search()*, *notify()*, *subscribe()*, *unsubscribe()*, *patch()*, *search()*, e *connect()*.



**INSTITUTO
FEDERAL**
Espírito Santo