

```
//UTILIZANDO EXPRESS PARA CRIAR UM SERVIDOR SIMPLES:
```

//Todo serviço WEB precisa de ter uma porta de comunicação entre o lado cliente e servidor - Só lembrando, utilizamos portas para identificar á qual serviço web estamos referenciando para que haja uma diferenciação entre um determinado serviço web e outro serviço, cada um deverá ter a sua própria porta.

```
const porta = 3003
```

```
//Importamos o framework express...
```

```
const express = require('express')
```

//Abaixo nós criamos uma instância para o framework express por através do módulo "express()", assim tornamos possível a utilização dos módulos do framework por através da variável que receberá essa instância, que no caso é "app"...

```
const app = express()
```

//Abaixo estamos usando o método "get()" de express, esse método é responsável por gerar uma resposta utilizando o método get invés do post, onde conseguiremos capturar os valores contidos no caminho "/produtosGet", porém note que não atribuímos valor nenhum para ser enviado quando o método get for chamado, em vez disso, nós ativamos uma função next() que por padrão irá realizar um middleware, chamando a próxima função...

```
app.get('/produtosGet', (request, response, next) => { //get só pode responder a uma pesquisa get, nunca poderá responder a uma pesquisa post...
```

```
    next()
```

```
})
```

//Podemos também utilizar o método "use()", esse método não irá procurar por um caminho específico get ou post, em vez disso ele vai executar o comando que estiver dentro de use()...

```
app.use((request, response, next) => {
```

```
    response.send({nome: 'Celular', preco: 123.46}) //O método send() irá enviar uma resposta a requisição http desse arquivo...
```

```
})
```

//Agora temos um exemplo usando o método post...

```
app.post('/produtosPost', (request, response, next) => { //agora ele responderá a chamadas post, mas não responderá a chamadas get...
```

```
    response.send({nome: 'Notebook', preco: 123.46}) //Detalhe importante sobre send(), ele automaticamente transforma um objeto em JSON...
```

```
})
```

//Abaixo estamos usando o método "listen()", esse método é responsável por ativar o servidor sobre uma determinada porta, veja que ele também recebe uma callback, que pode ser executada em conjunto com a abertura do servidor, nesse caso, nós imprimimos no console e informação de que o servidor está ativo, e qual é a porta em que ele está ativo...

```
app.listen(porta, () => {  
  console.log(`O servidor está ativo na porta ${porta}...`)  
})
```

//Faça o exercício de executar essa porta, e depois a procure, seja no POSTMAN ou no browser por através do caminho: <http://localhost:3003/produtos>, você vai ver que receberá como resposta o objeto de produtos, mas no formato JSON...

RESULTADO NO CONSOLE...

```
[Running] node  
"c:\Users\Almoxarifado\Documents\javascript\arquivos_das_aulas\Projetos_Back_End\Projeto_API_Express\src\Criando_Servidor_com_Express.js"  
O servidor está ativo na porta 3003...
```

RESULTADO NO POSTMAN PARA O MÉTODO GET...

The screenshot displays the Postman application interface. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Help'. Below it, a navigation bar includes 'Home', 'Workspaces', 'Reports', and 'Explore', along with a search bar and buttons for 'Sign In' and 'Create Account'. A yellow banner indicates 'Working locally in Scratch Pad. Switch to a Workspace'.

The main interface is divided into three sections:

- Left Panel (Scratch Pad):** Contains a sidebar with 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The main area shows a message: 'You don't have any collections. Collections let you group related requests, making them easier to access and run. Create Collection'.
- Top Right Panel:** Shows the request details: 'GET http://localhost:3003/...' and 'No Environment'.
- Right Panel:** Displays the request configuration and response.
 - Request:** Method 'GET', URL 'http://localhost:3003/produtosGet'.
 - Params:** Tabbed view showing 'Query Params' with a table:

KEY	VALUE	DESCRIPTION
Key	Value	Description
 - Response:** Tabbed view showing 'Body' with status '200 OK', time '23 ms', and size '268 B'. The response body is a JSON object:

```
{  "nome": "Celular",  "preco": 123.46}
```

The bottom of the image shows the Windows taskbar with various application icons and the system clock indicating 11:26 on 19/11/2021.

RESULTADO NO POSTMAN PARA O MÉTODO USE...

The screenshot displays the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', and 'Help'. Below it, there are tabs for 'Home', 'Workspaces', 'Reports', and 'Explore', along with a search bar and buttons for 'Sign In' and 'Create Account'. A yellow banner at the top indicates 'Working locally in Scratch Pad. Switch to a Workspace'.

The main interface is divided into three sections:

- Left Panel (Collections):** Shows a message 'You don't have any collections' with a 'Create Collection' link.
- Middle Panel (Request Editor):** Displays a GET request to 'http://localhost:3003/FAFAP'. The 'Params' tab is active, showing a table for Query Params.
- Right Panel (Response Viewer):** Shows the response body in JSON format, indicating a successful status (200 OK).

Query Params Table:

KEY	VALUE	DESCRIPTION
Key	Value	Description

Response Body (JSON):

```
{  "nome": "Celular",  "preco": 123.46}
```

The bottom status bar shows the 'Runner' tab, 'Trash' icon, and system information: 21°C, 11:27, 19/11/2021.

RESULTADO NO POSTMAN PARA O MÉTODO POST...

The screenshot displays the Postman application interface. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Help'. Below it, a navigation bar includes 'Home', 'Workspaces', 'Reports', and 'Explore', along with a search bar and buttons for 'Sign In' and 'Create Account'. A yellow banner indicates 'Working locally in Scratch Pad. Switch to a Workspace'.

The main interface is divided into three sections:

- Left Panel (Scratch Pad):** Contains a sidebar with 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The main area shows a message: 'You don't have any collections. Collections let you group related requests, making them easier to access and run. Create Collection'.
- Top Right Panel:** Shows the request details: 'POST http://localhost:3003/produtosPost'. It includes a 'Send' button and tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'.
- Bottom Panel:** Displays the response body in 'Pretty' format, showing a JSON object:

```
{  "nome": "Notebook",  "preco": 123.46}
```

. It also shows the status '200 OK', time '18 ms', and size '269 B'.

The Windows taskbar at the bottom shows the search bar with the text 'Digite aqui para pesquisar' and various application icons. The system tray on the right indicates the date and time as '11:29 19/11/2021'.