Desestruturação de Objetos

O que já fizemos?

```
const { titulo, descricao } = req.body;
```

Forma mais curta, legível e elegante de **extrair valores de um objeto**

Vamos considerar o seguinte objeto:

```
const pessoa = {
  nome: "Angelo",
  idade: 24,
  email: "hankangelo6@gmail.com",
  endereco: "Rua tal",
};
```

Como podemos acessar os valores de dentro dele? O nome, por exemplo.

Eu poderia escrever algo assim

```
let nome = pessoa.nome;
let email = pessoa.email;
let idade = pessoa.idade;

console.log(`nome: ${nome}, email: ${email}, idade: ${idade}`)
```

Mas, com a desestruturação…

```
const { nome, email, idade } = pessoa;
```

“Crie 3 variaveis, uma chamada "nome", outra "email", e "idade", e atribua pra elas os valores correspondentes de dentro do objeto PESSOA”

O que acontece se mudarmos a chave de nome?

```
const pessoa = {
  name: "Angelo",
  age: 24,
  email: "hankangelo6@gmail.com",
  adress: "Rua tal",
};
```

Eu vou conseguir pegar, automaticamente, a propriedade “nome”, “idade”, “email”, etc?

```
const { nome, email, idade } = pessoa;
```

Aplicando isso a REQUESTS

```
app.post("/", (req, res) => {
  //quero pegar um dado que eu sei que vai vir como "nome" do meu BODY
  let nome = req.body.nome;
  console.log(nome);
});
```

```
app.post("/", (req, res) => {
  //quero pegar um dado que eu sei que vai vir como "nome" do meu BODY
  const { nome } = req.body;
  console.log(nome);
});
```

Fixando

- Acesse o repositório da matéria
- Procure pela pasta de arquivos auxiliares
- Copie o conteúdo do arquivo chamado *exercicios-desestruturacao.js*
- crie um novo arquivo na sua máquina (com a extensão .js)
- cole o conteúdo que copiou
- Realize os exercícios propostos
- Envie o arquivo (com a extensão correta) para "hankangelo6@gmail.com"
- No assunto do email escreva: nome + exercicio da aula de desestruturação

Vamos fazer juntos

Exercício 1

Você tem o seguinte objeto

```
const aluno = {
  nome: "Joana",
  idade: 21,
  curso: "Engenharia",
};
```

Use desestruturação para criar as variáveis “nome” e “curso” e imprima a frase:

“Joana está cursando engenharia”

Para escrever uma frase usando valores das variáveis, você pode usar o seguinte formato:

`{nome} está cursando ${curso}`

Exercício 2 - Desestruturando um objeto dentro de uma função

Crie uma função que recebe um objeto chamado “produto”, que terá a seguinte definição:

```
const produto = {
  nome: "Celular",
  preco: 1500,
  categoria: "eletronicos",
};
```

A função deve usar **desestruturação** para extrair os dados e retornar uma mensagem no seguinte formato:

- A desestruturação deve ser feita direta nos parâmetros da função

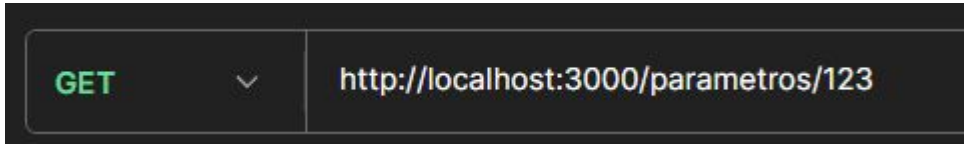
“Produto: celular, Categoria: eletronico, preco: R$1500”

`{variavel} texto que voce quer ${variavel}`

Query Params

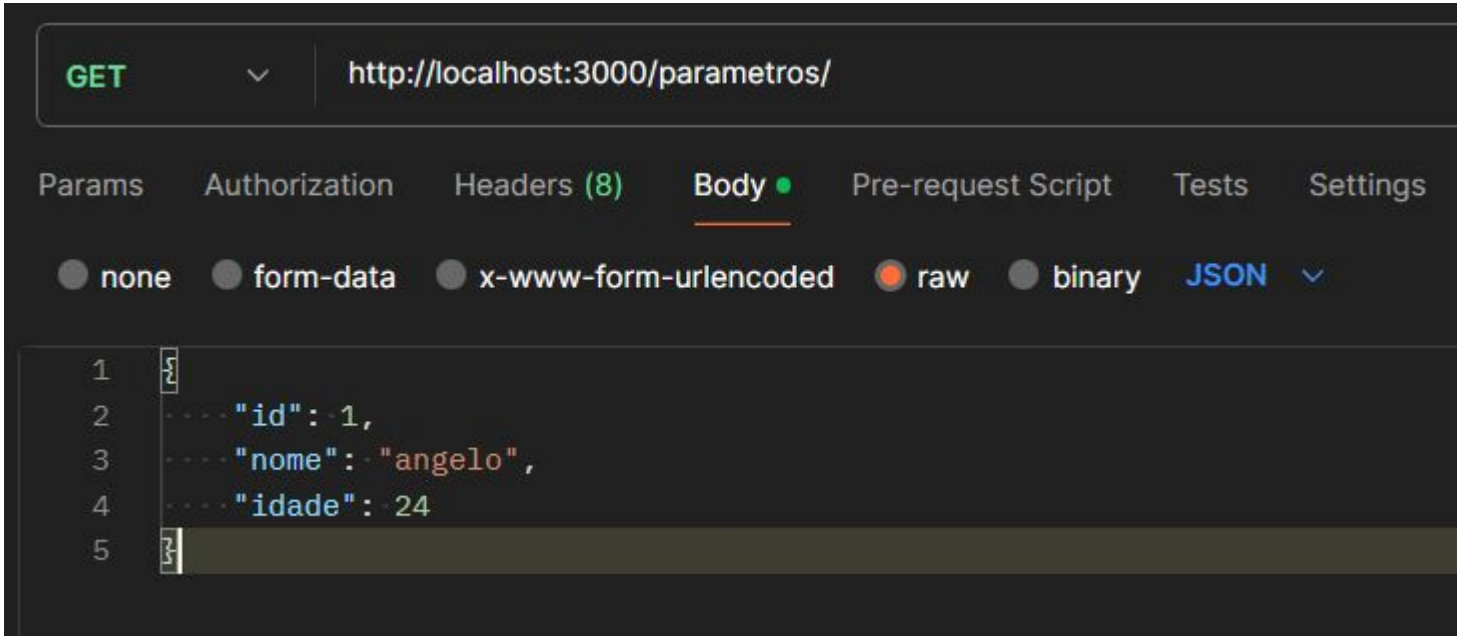
Até aqui já vimos…

Path params




```
router.get("/:id", (req, res) => {
  imprime(`chamando rota GET -> listar apenas um. ID: ${req.params.id}`);
  listarUm(req, res);
});
```

Body


GET
http://localhost:3000/parametros/
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
JSON
1
2 "id": 1,
3 "nome": "angelo",
4 "idade": 24
5

O que é o Query Params?

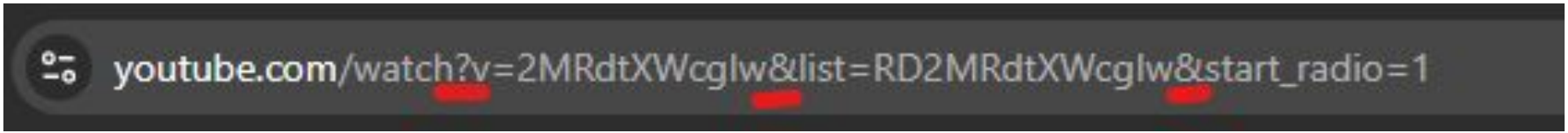
Parâmetros de consulta

Valores enviados na URL, após o símbolo “**?**”.

Assim como o body e path params, é usado para passar informações para a requisição.

Formato: *http://localhost:3000/rota?chave=valor*

Exemplo: *http://localhost:3000/usuarios?nome=angelo*



https://www.youtube.com/watch**?**v=2MRdtXWcgIw&list=RD2MRdtXWcgIw&start_radio=1

Para que serve o Query Params?

- Filtros
- Ordenações
- Paginação
- Busca

Modifica o comportamento da resposta sem precisar mudar a rota

Exemplos

/produtos?categoria=eletronicos

/postagens?autor=joao&limit=5

/tarefas?status=done

Os parâmetros, passados dessa maneira, permitem que o cliente personalize o resultado, **sem precisar alterar a estrutura da API**

Exemplo real

Beecrowd

https://judge.beecrowd.com/pt/problems/index/1?page=2

Mas afinal…

Qual a diferença disso para o params como já conhecemos?

Query Params

- Aparecem após o “?” na URL
- Opcionais
- Usado para filtros, buscas, paginação
- acessa com *req.query*
- não precisa de alteração na API

Path Params

- aparece logo depois da rota
- geralmente obrigatório
- Usado para identificar recursos específicos (usuários, por exemplo)
- acessa com *req.params*
- Precisa de alteração na API

Fixando

- Acesse o repositório da matéria
- Procure pela pasta de arquivos auxiliares
- Copie o conteúdo do arquivo chamado *exercicios-query-params.js*
- crie um novo arquivo na sua máquina (com a extensão .js)
- cole o conteúdo que copiou
- Realize os exercícios propostos
- Envie o arquivo (com a extensão correta) para "hankangelo6@gmail.com"
- No assunto do email escreva: nome + exercicio da aula de query-params

Vamos fazer juntos