

ResponseEntity é uma classe utilitária do Spring Web para auxiliar na resposta HTTP. Representa o tipo de resposta que vai ser retornada.

Método para buscar um Produto

```
@GetMapping("/{id}")
public ResponseEntity<Produto> pesquisar(@PathVariable Long id) {
    Optional<Produto> produto = produtoRepository.findById(id);
    if (produto.isPresent()) {
        return ResponseEntity.ok(produto.get());
    }

    return ResponseEntity.notFound().build();
}
```

Se existir produto vamos retornar usando o método **get** que busca o que está armazenado no **Optional**, caso não existir retorna 404, o método build retorna um **ResponseEntity**.

Testando no Postman com um código existente

GET http://localhost:8080/produtos/2

Status: 200 OK Time: 80 ms Size: 235 B Save Response

Body

```
1 {
2   "id": 2,
3   "descricao": "Kindle",
4   "valor": 200.0,
5   "dataCadastro": "2021-02-18"
}
```

Testando no Postman com um código inexistente

GET http://localhost:8080/produtos/3332

Status: 404 Not Found Time: 17 ms Size: 130 B Save Response

Body

```
1
```

Código	ResponseEntity Method
Padrões	<code>new ResponseEntity<>(HttpStatus.<STATUS>)</code> <code>ResponseEntity.status(HttpStatus.CREATED).build()</code> <code>new ResponseEntity<>(objeto, HttpStatus.<STATUS>)</code> <code>ResponseEntity.status(HttpStatus.CREATED).body(objeto)</code>
200 - Ok	<code>ResponseEntity.ok().build()</code> <code>ResponseEntity.ok(objeto)</code>
201 - Criado HttpStatus.CREATED	<i><code>ResponseEntity.created(uri).body(objeto) *</code></i>
204 - Sem Conteúdo	<code>ResponseEntity.noContent().build()</code>
400 - Bad Request	<code>ResponseEntity.badRequest().build()</code> <code>ResponseEntity.badRequest().body(objeto)</code>
404 - não encontrado	<code>ResponseEntity.notFound().build()</code>

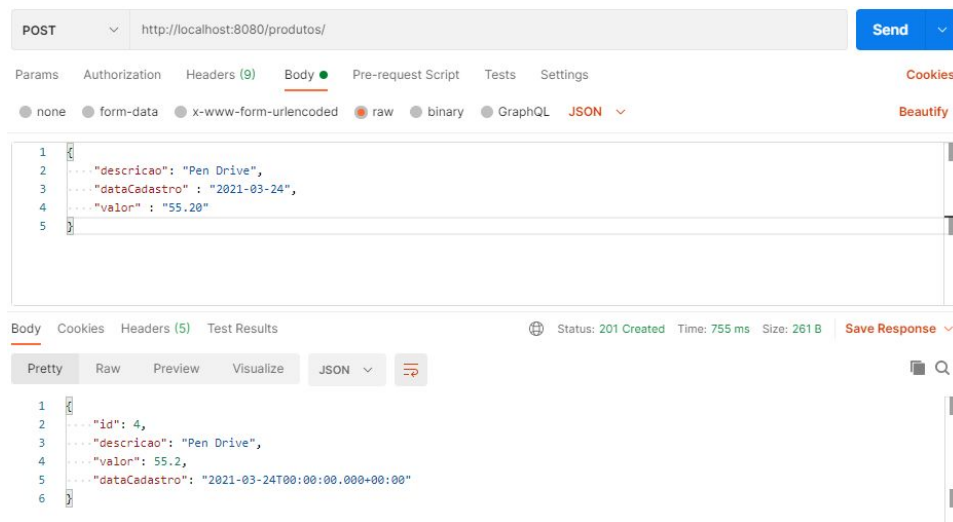
Método `created` com parâmetro `uri` deve ser usado com *HATEOAS* tutorial aqui <https://www.baeldung.com/spring-hateoas-tutorial>
Mais informações sobre os códigos http <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status>

Adicionado um Produto no banco de dados.

```
@PostMapping
@ResponseStatus(HttpStatus.CREATED)
public Produto inserir(@RequestBody Produto produto) {
    return produtoRepository.save(produto);
}
```

@RequestBody - Transformando JSON em um objeto Produto

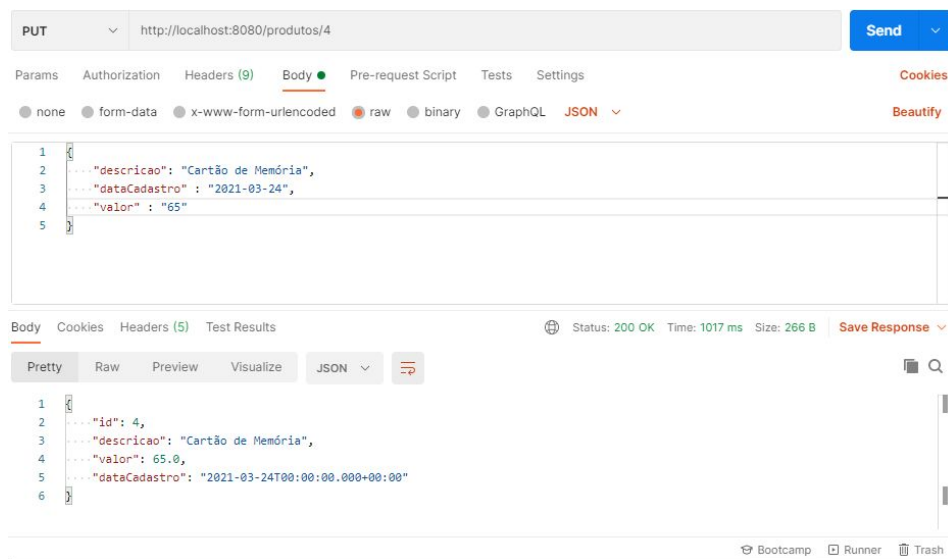
Testando no Postman



Alterando um Produto no banco de dados.

```
@PostMapping("/{id}")
public ResponseEntity<Produto> atualizar(@PathVariable Long id, @RequestBody Produto produto) {
    if (!produtoRepository.existsById(id)) {
        return ResponseEntity.notFound().build();
    }
    produto.setId(id);
    produto = produtoRepository.save(produto);
    return ResponseEntity.ok(produto);
}
```

Atribuímos o **id** ao **setId** pois senão seria criado um novo registro na tabela porque o id vem nulo, não é definido no corpo da requisição. No retorno do método **save**, vamos atribuir o valor à variável **produto**.



The screenshot shows a REST client interface with a PUT request to `http://localhost:8080/produtos/4`. The request body is a JSON object: `{ "descricao": "Cartão de Memória", "dataCadastro": "2021-03-24", "valor": "65" }`. The response status is 200 OK, and the response body is a JSON object: `{ "id": 4, "descricao": "Cartão de Memória", "valor": 65.0, "dataCadastro": "2021-03-24T00:00:00.000+00:00" }`.

PUT `http://localhost:8080/produtos/4` Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   ... "descricao": "Cartão de Memória",
3   ... "dataCadastro": "2021-03-24",
4   ... "valor": "65"
5 }
```

Body Cookies Headers (5) Test Results Status: 200 OK Time: 1017 ms Size: 266 B Save Response

Pretty Raw Preview Visualize **JSON** 🔍

```
1 {
2   ... "id": 4,
3   ... "descricao": "Cartão de Memória",
4   ... "valor": 65.0,
5   ... "dataCadastro": "2021-03-24T00:00:00.000+00:00"
6 }
```

🗑 Bootcamp 🏠 Runner 🗑 Trash

Excluindo um Produto no banco de dados.

```
@DeleteMapping("/{id}")
public ResponseEntity<Void> remover(@PathVariable Long id) {
    if (!produtoRepository.existsById(id)) {
        return ResponseEntity.notFound().build();
    }
    produtoRepository.deleteById(id);
    return ResponseEntity.noContent().build();
}
```

DELETE http://localhost:8080/produtos/4 Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (3) Test Results

Status: 204 No Content Time: 592 ms Size: 112 B Save Response

Pretty Raw Preview Visualize Text

Não retornamos corpo só indicamos que deu certo com o código **204** que é um código mais específico para esta situação

Ao Tentarmos excluir novamente o id 4 qual código deverá ser retornado?