



Luís Ignacio

# API Rest com .net 7.0



# Tópicos

- Apresentação
- O que é API?
- O que é REST?
- Princípios do REST
- HTTP e REST
- Definindo recursos e endpoints
- Métodos HTTP
- Retornando respostas HTTP
- Manipulando dados com JSON
- Trabalhando com verbos HTTP
- Implementando o projeto

# Luís Ignacio

- Tecnólogo em Análise e Desenvolvimento de Sistemas - **ULBRA** - Torres / RS
- 34 anos
- Desenvolvedor backend, Techlead C# e Gestor de Cloud na **Dragon Venture Capital**



[Voltar ao slide de tópicos](#)

# O que é API?

Application Programming Interface (Interface de Programação de Aplicativos)

**API** é uma interface que permite a **comunicação** entre diferentes softwares, facilitando a **integração** e o **compartilhamento** de informações. Ela define regras e protocolos para que os sistemas possam interagir de **maneira padronizada**. As APIs são amplamente utilizadas para **integração de serviços**, **desenvolvimento de aplicativos** e **acesso a dados**.

[Voltar ao slide de tópicos](#)



# O que é REST?

**R**epresentational **S**tate **T**ransfer, é um estilo arquitetural utilizado para projetar sistemas **distribuídos** na web. Ele define um **conjunto de princípios** para criar serviços web **escaláveis, interoperáveis** e de fácil **manutenção**.



[Voltar ao slide de tópicos](#)

# Princípios do REST



- Utilização de uma interface **uniforme** para interações entre **cliente** e **servidor**;
- Recursos são identificados por **URLs** (Uniform Resource Locators);
- Uso de métodos HTTP (**GET, POST, PUT, DELETE**) para operar nos recursos;
- **Stateless**: cada solicitação do cliente **deve conter todas as informações necessárias** para o servidor entender e processar a requisição.

[Voltar ao slide de tópicos](#)

# HTTP e REST

O protocolo **HTTP** (Hypertext Transfer Protocol) é a base para implementar serviços **RESTful**. Ele define métodos, cabeçalhos e códigos de status que são usados para interagir com os recursos.

A diferença entre **HTTP** e **HTTPS** (Hypertext Transfer Protocol Secure) está relacionada à **segurança** e **criptografia** dos dados transmitidos entre um cliente (navegador) e um servidor.



# Definindo recursos e endpoints

[Voltar ao slide de tópicos](#)

Um **recurso** é uma entidade no sistema que pode ser acessada através da API. Cada recurso deve ter uma URL única.

Os **endpoints** são as URLs que representam os recursos e são usados para realizar operações neles. São nos endpoints que as requisições fazem o primeiro contato com a API.



# Métodos HTTP

[Voltar ao slide de tópicos](#)

Os **métodos HTTP** indicam a operação que será realizada no recurso.

Os principais métodos são:

- **GET**: Recupera um recurso ou uma lista de recursos;
- **POST**: Cria um novo recurso;
- **PUT**: Atualiza um recurso existente;
- **DELETE**: Remove um recurso.

# Retornando respostas HTTP

Ao lidar com uma solicitação HTTP, a API deve retornar uma resposta apropriada. Isso inclui o **código de status HTTP**, **cabeçalhos** e, opcionalmente, um **corpo de resposta**.

Exemplos de códigos de status são:

- 200 OK
- 201 Created
- 400 Bad Request
- 404 Not Found

entre outros.

[Voltar ao slide de tópicos](#)

# Manipulando dados com JSON

JSON (JavaScript Object Notation) é um formato de dados amplamente utilizado em APIs RESTful para representar **objetos** e suas **propriedades**. No .NET Core, você pode usar as bibliotecas JSON.NET ou System.Text.Json para **serializar** e **desserializar** objetos JSON.

[Voltar ao slide de tópicos](#)

# Trabalhando com verbos HTTP

No .NET Core, você pode usar **atributos** para associar os verbos HTTP aos métodos de ação nos controladores.

Por exemplo:

- O atributo **[HttpGet]** associa um método ao verbo **GET**.
- O atributo **[HttpGet("itens")]** associa um método ao verbo GET em um endpoint **"/itens"**

[Voltar ao slide de tópicos](#)

# Implementando o projeto

É hora de pôr a mão na massa :)



Luís Ignacio

@luishti

[linkedin.com/in/luishtignacio](https://linkedin.com/in/luishtignacio)

[Voltar ao slide de tópicos](#)