# API e REST

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

# AGENDA

- API
- Padrão REST
- Constraints
- Modelos de Maturidade Richardison
- Recursos
- API REST
- CRUD
- Método HTTP
- Código de Status de Resposta HTTP
- Endpoint
- Projeto Pré-Projeto

# API

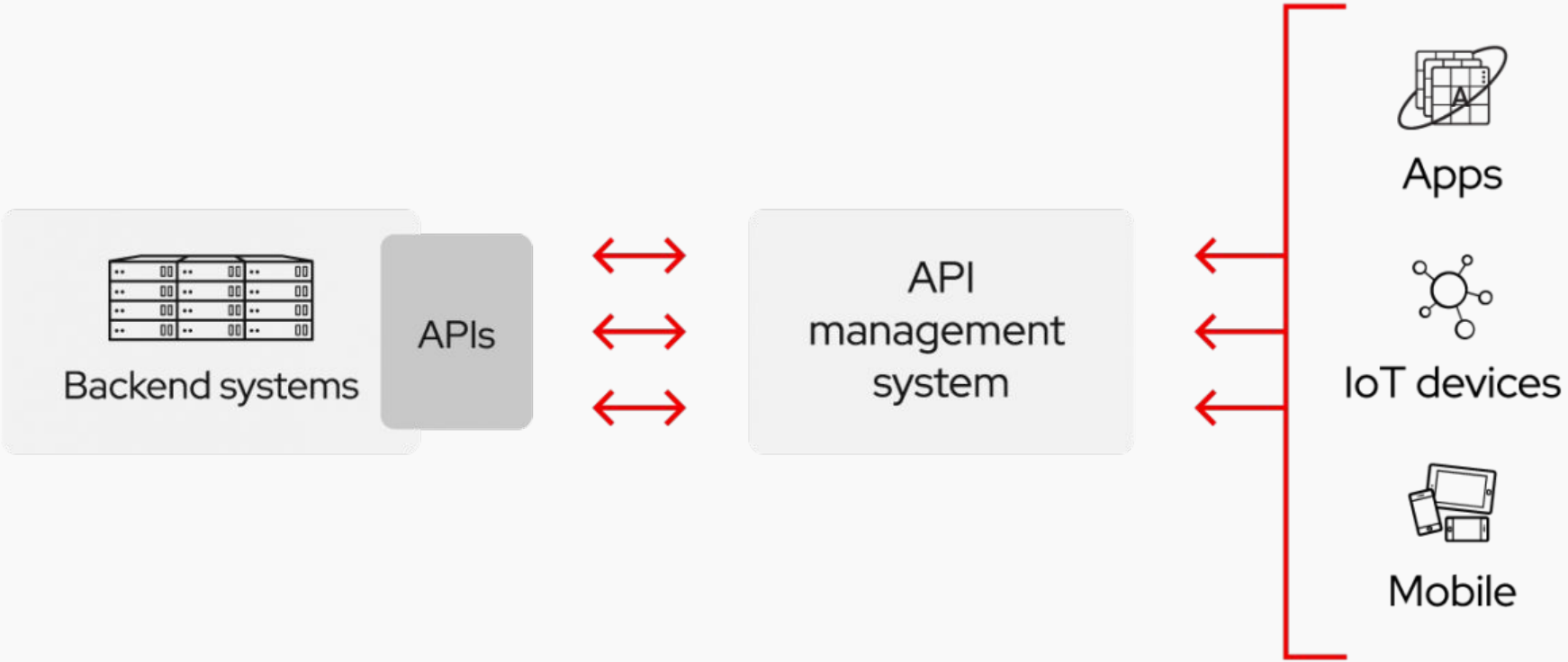
**API - Application Programming Interface(Interface de programação de aplicação)**

Uma API atua como um contrato entre 2 partes, uma realiza a chamada e outra gera a resposta ou realiza uma ação.

Apis tem algumas características:

- São interfaces para que um sistema possa utilizar outro sistema.
- Podem ser utilizadas em sistemas locais, como o Sistema Operacional, onde o SO(Sistema Operacional) disponibiliza um conjunto de APIs para utilizar o sistema.
- No caso de um sistema web a API é a responsável pela lógica do sistema, ou pode realizar uma tarefa específica.
- Normalmente se comunica pela internet através do padrão REST.

# API


Backend systems
APIs
API management system
Apps
IoT devices
Mobile

# Padrão REST

**REST - Representational State Transfer (Transferência de estado representacional)**

O REST é um termo cunhado em uma tese de Ph.D de Roy Fielding, que também foi um dos autores da especificação do protocolo HTTP.

Essa tese tinha como objetivo a formalização de um conjunto de melhores práticas, as constraints(restrições), que são padrões para o uso do HTTP e da URI de forma a utilizar todos os seus recursos.

REST é um estilo de arquitetura. Isso significa que não há um padrão oficial para APIs RESTful web. Conforme definido na dissertação de Roy Fielding “Architectural Styles and the Design of Network-based Software Architectures”

Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST) (uci.edu)

# Constraints

- Separação Cliente-Servidor - escalabilidade independente.
- Stateless - requisições independentes umas das outras, sem transferência de estado.
- Cache - armazenamento temporário de respostas em cache.
- Interface uniforme - Cada recurso deve ter apenas uma forma de ser acessado e também deve ser padronizado em nomenclatura e formato de links. Recursos, Mensagens Autodescritivas, Hypermedia.
- Sistema em camadas - capacidade de adicionar sistemas intermediários de forma transparente para o cliente.
- Código sob demanda (opcional) - retorno de código executável. É incomum de ser usado.
- O JSON e o XML são representações de dados, que são utilizadas para trafegar os dados dentro das especificações REST. É o formato da informação.

# Modelos de Maturidade Richardison

- **Nível 0 - POX**
  - Não há padrão para os endpoints dos recursos, e não há regra implementada com base nos métodos HTTP.
- **Nível 1 - Recursos**
  - Uso de substantivo no plural para representar os recursos no sistema, uso dos verbos HTTP
- **Nível 2 - Verbos HTTP**
  - Uso de GET, POST, PUT, DELETE, métodos idempotentes, onde o método pode ser executado várias vezes sem alterar os estado do servidor
- **Nível 3 - HATEOAS**
  - Hypermedia as the Engine of Application State.
  - Implementa links que indicarão outros recursos ligados ao pedido.

# Modelos de Maturidade Richardison

- **Nível 3 - HATEOAS**

```
{
    "cursos": [
        {
            "id": 1,
            "nome": "C# (C Sharp)",
            "links": [
                {
                    "type": "GET",
                    "rel": "self",
                    "uri": "api.treinaweb.com.br/cursos/1"
                },
                {
                    "type": "GET",
                    "rel": "curso_aulas",
                    "uri": "api.treinaweb.com.br/cursos/1/aulas"
                }
            ]
        }

    ]

}
```

# Recursos

**Resources(Recursos):** elementos de informação que podem ser manipulados por meio de um identificador global. São sempre substantivos, nunca verbos.

**URI(Uniform Resource Identifier):** uma cadeia de caracteres usada para identificar um recurso na internet

- Recurso: Usuário
- uri: www.site.com/usuario
- uri: www.site.com/user

**URL(Uniform Resource Locator):** endereço do recurso na rede.

# INTERVALO DE AULA

**DEV!**

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

**Início:** 20:20
**Retorno:** 20:40

# CRUD

O CRUD é um acrônimo para Create Read Update e Delete, que são operações que APIs Web realizam de forma geral. Sendo assim utilizamos o protocolo HTTP e seus métodos para recebermos chamadas em um programa e gerarmos respostas e/ou armazenar informações.

Essa API deve seguir no mínimo o nível 2 do modelo de Richardson e deve tratar erros e deve retornar as respostas no método correto do REST. Geralmente essa API se comunicará pelo padrão JSON.

# API REST

API REST - Padrão para troca de informação entre aplicações, APIs desenvolvidas sobre os conceitos REST. Elas utilizam o protocolo HTTP(1.1 ou maior) e faz uso dos métodos:

Os métodos HTTP, ou verbos HTTP, são um conjunto de ações que podem ser executadas através do protocolo HTTP.

**GET:**

- O método GET solicita a representação de um recurso específico. Requisições utilizando o método GET devem retornar apenas dados.

**POST:**

- O método POST é utilizado para submeter uma entidade a um recurso específico, frequentemente causando uma mudança no estado do recurso ou efeitos colaterais no servidor.

**PUT:**

- O método PUT substitui todas as atuais representações do recurso de destino pela carga de dados da requisição.

# Métodos HTTP

**DELETE:**

- O método DELETE remove um recurso específico.

**Outros Métodos:**

- PATCH - O método PATCH é utilizado para aplicar modificações parciais em um recurso.
- CONNECT - estabelece um túnel para o servidor identificado pelo recurso de destino.
- OPTIONS - é usado para descrever as opções de comunicação com o recurso de destino.
- TRACE - executa um teste de chamada loopback junto com o caminho para o recurso de destino.
- HEAD - solicita uma resposta de forma idêntica ao método GET, porém sem conter o corpo da resposta.

# Código de Status de Resposta HTTP

Os status de retorno HTTP demonstram o resultado de uma chamada HTTP, sendo assim temos diversos tipos de retorno que podem ocorrem quando chamamos uma API.

Alguns dos principais códigos de retorno do protocolo HTTP:

- 200 -> OK
- 201 -> Created
- 204 -> No content
- 401 -> Unauthorized
- 404 -> Not found
- 405 -> Method not allowed
- 500 -> Internal Server error

Códigos de status de respostas HTTP - HTTP | MDN (mozilla.org)

# Endpoints

Os Endpoints são a extremidade de um canal de comunicação, por exemplo, um URL. Eles são formas de acessarmos as funcionalidades de uma API.

Elá está dividida em ABC:

- A - Address (Endereço). localhost:8080; google.com
- B - Binding (Acesso). localhost:8080/pessoa
- C - Contract (Contrato). localhost:8080/pessoa {“nome”:””, “idade”:””}

# Projeto Pré-Projeto

Vamos fazer durante a semana um projeto para gerar questionários de forma automática
Abaixo temos um exemplo que podemos utilizar para criar o projeto:

andresantnunes/notas-exemplo (github.com)

# AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

# DEVinHouse

Parcerias para desenvolver a sua carreira

## OBRIGADO!

SENAI

<LAB365>