Boas-vindas!

Olá, Mentoramer!
Parabéns! Você está avançando cada vez mais no seu desenvolvimento em **JAVA!**

Só para lembrá-lo, no curso PRO também temos os **guias de estudo** para cada um dos módulos. Como você sabe, nesses guias você encontrará um pequeno resumo do que aprendeu durante as aulas e dicas extras para evoluir ainda mais!

Bom, não se esqueça que você pode **tirar mais dúvidas e iniciar discussões produtivas com os nossos mentores lá na nossa comunidade no Discord.**

Aproveite a sua jornada de aprendizagem ao máximo e conte com o **apoio completo do time Mentorama**!

Módulo #1

Nesse módulo você teve o **primeiro contato com desenvolvimento web**. Há alguns anos, o desenvolvimento web parou de se limitar à criação de sites, passando por uma evolução que permitiu que até mesmo programadores de outras linguagens que não HTML, CSS e JavaScript pudessem criar sites e aplicações inteiras com base na web.

Vamos rever alguns conceitos aprendidos nesse módulo.

APIs

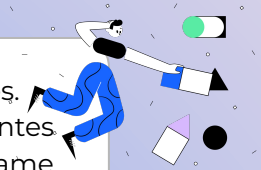
APIs (*Application Programming Interface*) são programas que permitem que outras aplicações utilizem suas funcionalidades para estender alguma parte dela mesma.

Por exemplo:

- Um sistema de vendas normalmente deve saber encontrar um endereço completo com base no CEP informado pelo cliente. Esse sistema pode usar a API dos Correios para encontrar o logradouro, bairro, cidade e estado de um CEP. É exatamente isso que a API do ViaCEP (https://viacep.com.br/) faz. Experimente acessar esse endereço no seu navegador: https://viacep.com.br/ws/01001000/json/

- Um e-commerce também precisa saber cobrar um cliente com base na forma de pagamento escolhida pelo mesmo. Não é necessário desenvolver uma parte de cobrança do zero para fazer isso. Basta integrar APIs de sistemas de pagamentos como Pagar.me (https://pagar.me/), Stripe (https://stripe.com/), entre outros ao seu e-commerce.

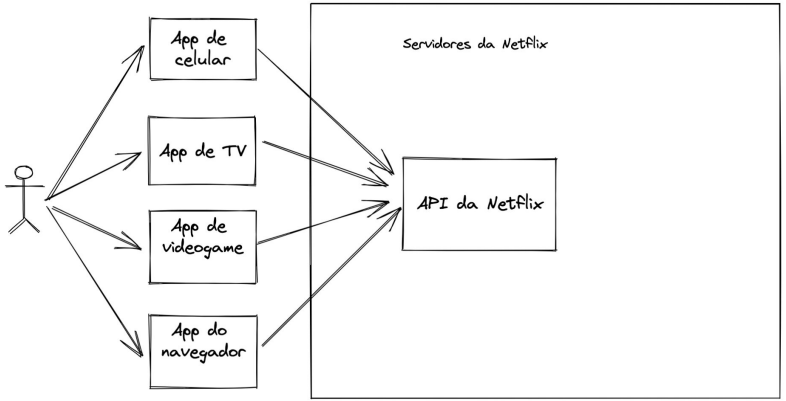
No contexto de desenvolvimento web as APIs são programas escritos em uma linguagem de programação qualquer que responde a requisições HTTP de outros programas e sites.



As APIs não servem apenas para estender funcionalidades. Servem também para criar aplicações inteiras para diferentes plataformas (web, celular, TV, desktop, console de videogame, etc...).

Tenhamos como exemplo a **Netflix**, que é um serviço que podemos acessar em diferentes plataformas, seja no navegador, celular, TV, videogame, entre outros. Os desenvolvedores da Netflix não tiveram que programar toda a lógica do serviço em cada uma dessas plataformas. Ao invés disso, eles criaram uma API que fornece todo o catálogo de filmes e séries, gestão de perfis, cobrança, entre outros, então as aplicações de cada plataforma apenas se conectam com essas API para fornecer a melhor experiência possível para seus usuários.

Se você quiser aprender muito mais sobre como funciona a engenharia de software por trás da Netflix, recomendamos que acesse o Blog Técnico da empresa: https://netflixtechblog.com/



Spring Framework

O **Spring Framework** (https://spring.io/) é o principal framework para desenvolvimento web usado no ecossistema Java.
É completo, oferecendo suporte para desenvolvimento de APIs, sites e outros tipos de programas.

O Spring é o que chamamos de projeto Umbrella (guarda-chuva), pois é composto por diversos projetos especializados. Os principais deles são:

- **Spring Data:** framework que oferece diversas classes que facilitam o acesso e gerenciamento de banco de dados;

- **Spring Security**: permite facilmente configurar uma camada de segurança em uma aplicação, fornecendo sistema de autenticação, autorização, controle de privilégios, além de permitir se conectar a diferentes meios de autenticação.

Verbos HTTP

Quando desenvolvemos uma API web, nós fornecemos diversas interfaces que permitem que outras aplicações interajam com ela por meio de endereços.

Vamos considerar uma API que permite fazer a gestão de clientes. Essa API permite:

- Cadastrar um cliente;
- Atualizar um cliente;
- Obter dados de um cliente com base em seu ID;
- Obter todos os clientes cadastrados;
- Apagar um cliente com base em seu ID;

Para essa API desenvolvemos as seguintes rotas:

- Cadastro: `POST /api/clientes/`
- Atualização `PATCH /api/clientes/:id`
- Obter dados de um cliente: `GET /api/clientes/:id`
- Obter dados de todos os clientes: `GET /api/clientes`
- Apagar um cliente: `DELETE /api/clientes/:id`

Note que antes do endereço, nós informamos também o verbo (ou métodos) HTTP. Esses verbos são usados para informar qual ação queríamos fazer com o recurso que estamos lidando.
O recurso, no caso, é o "cliente".

- **POST**: estamos criando um recurso em nosso sistema (estamos criando um cliente);
- **PATCH**: estamos atualizando dados de um recurso (atualizando dados do cliente);
- **GET**: estamos obtendo dados de um recurso: nesse caso definimos duas rotas que usam o verbo GET, uma para obter dados de um cliente com base em seu ID e outra para obter todos os clientes cadastrados;
- **DELETE**: estamos apagando dados de um recurso (apagando dados de um cliente com base em seu ID).

Existem muitos outros verbos HTTP que são poucos usados, mas que também servem para semanticamente definir a ação a ser executada em um recurso.

Recomendamos essa documentação da **Mozilla** que explica detalhadamente cada um deles:
https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods

Códigos de Status HTTP

Analogamente aos verbos HTTP, os códigos de status servem para informar qual foi o resultado do processamento de uma requisição. Vejamos os principais **códigos de status**:

- Respostas de informação (**100-199**);
- Respostas de sucesso (**200-299**);
  - **200** (**OK**): a requisição foi processada corretamente;
  - **201** (**Created**): o recurso foi criado corretamente
- Redirecionamentos (**300-399**);
- Erros do cliente (**400-499**);
  - **404** (**Not Found**): o recurso ou página não foi encontrada
- Erros do servidor (**500-599**);
  - **500** (**Internal Server Error**): ocorreu um erro interno no servidor.

Para saber sobre todos os códigos de status HTTP, recomendamos essa página da Mozilla que especifica cada um deles:
https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status

Postman

Nós podemos testar nossas APIs web usando programas que fazem requisições e permitem definir o verbo, parâmetros, corpo da requisição, etc.

O Postman (https://www.postman.com/) é uma dessas ferramentas.