UNIVERSIDADE FEDERAL DO CEARÁ - Campus Quixadá
Cursos: SI, ES, RC, CC e EC
Código: QXD0043
Disciplina: Sistemas Distribuídos

Capítulo 9 – Application Programming Interface (API)

Prof. Rafael Braga

Agenda

1. 1. Introdução;
2. Conceitos Básicos;
3. Funcionamento de uma API;
4. Vantagens das APIs;
5. Desenvolvimento de APIs;
6. Exemplos de APIs;
7. Conclusão.

Introdução

- Definição: O que é uma API?
  - Uma API (*Application Programming Interface*) é um conjunto de regras e definições que permite que diferentes sistemas de software se comuniquem entre si. Ela define os métodos e formatos que um programa pode usar para solicitar e trocar dados com outro sistema.

Introdução

- Por que as APIs são **importantes** em sistemas distribuídos?
  - As APIs são fundamentais para a integração de sistemas distribuídos, pois permitem que diferentes aplicativos, plataformas e serviços interajam de forma padronizada e segura. Elas facilitam a automação de processos, a interoperabilidade entre aplicações e a escalabilidade dos sistemas.

Conceitos Básicos

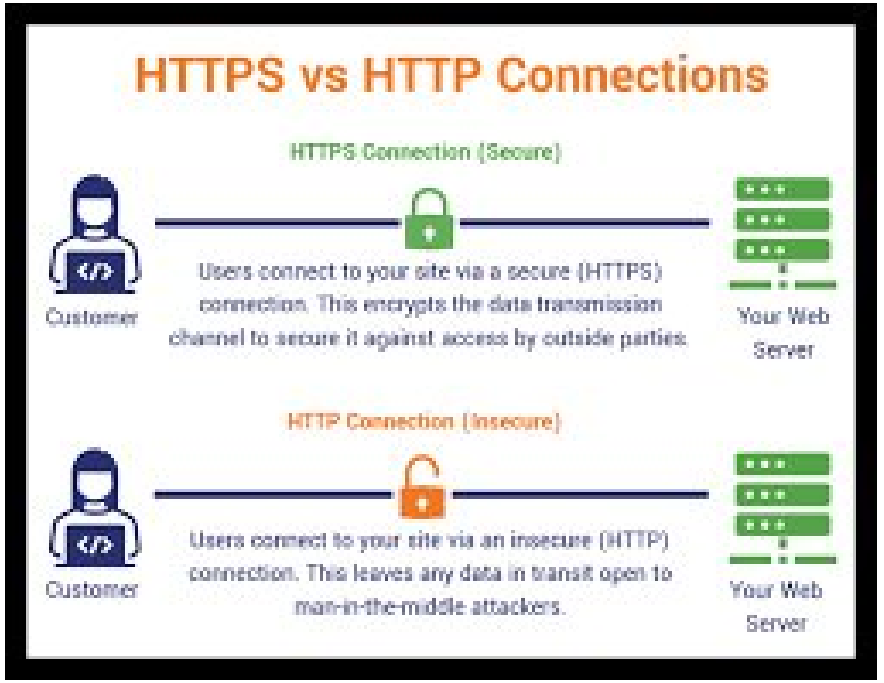
- Componentes de uma API:
  - Endpoints: URLs que representam pontos de acesso à API;
  - Métodos HTTP: GET (recuperação de dados), POST (criação de dados), PUT (atualização de dados), DELETE (remoção de dados);
  - Parâmetros: Informações passadas na URL ou no corpo da requisição;
  - Headers: Metadados enviados junto à requisição, como autenticação e tipo de conteúdo.
  - Corpo da requisição: Dados enviados no formato JSON, XML ou outro padrão.

Conceitos Básicos

- Tipos de APIs:
  - REST (Representational State Transfer): Baseado em HTTP, simples e amplamente utilizado;
  - SOAP (Simple Object Access Protocol): Protocolo mais rígido baseado em XML.
  - GraphQL: Permite consultas flexíveis e retorna apenas os dados necessários.
  - gRPC: Baseado em HTTP/2, utiliza protocolos binários para maior eficiência.

Conceitos Básicos

- Tipos de APIs:
  - HTTP/HTTPS: Padrão para a maioria das APIs web;
  - WebSocket: Comunicação bidirecional em tempo real;
  - MQTT: Protocolo eficiente para IoT;



Conceitos Básicos

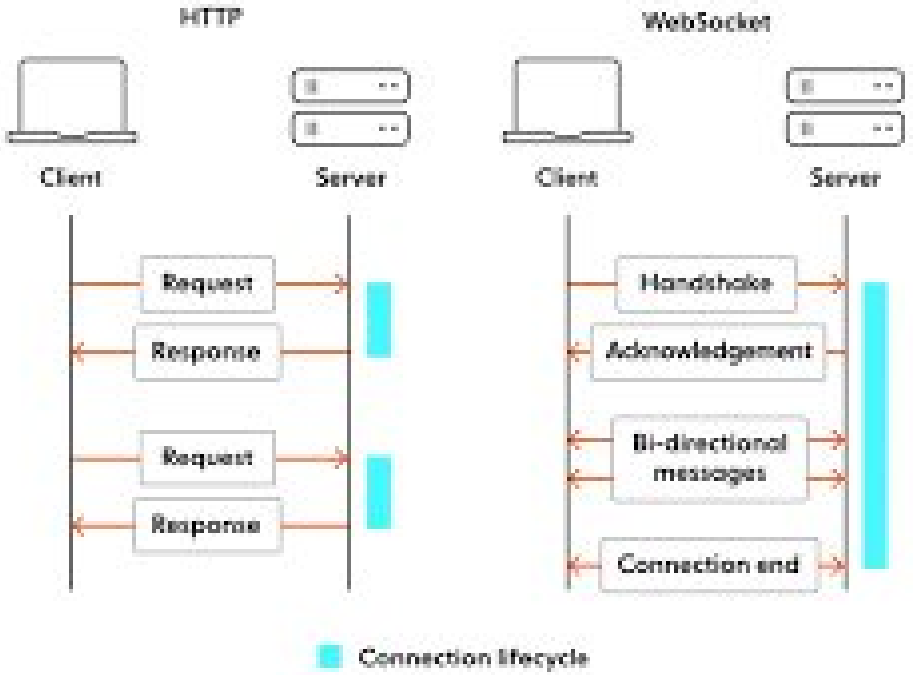
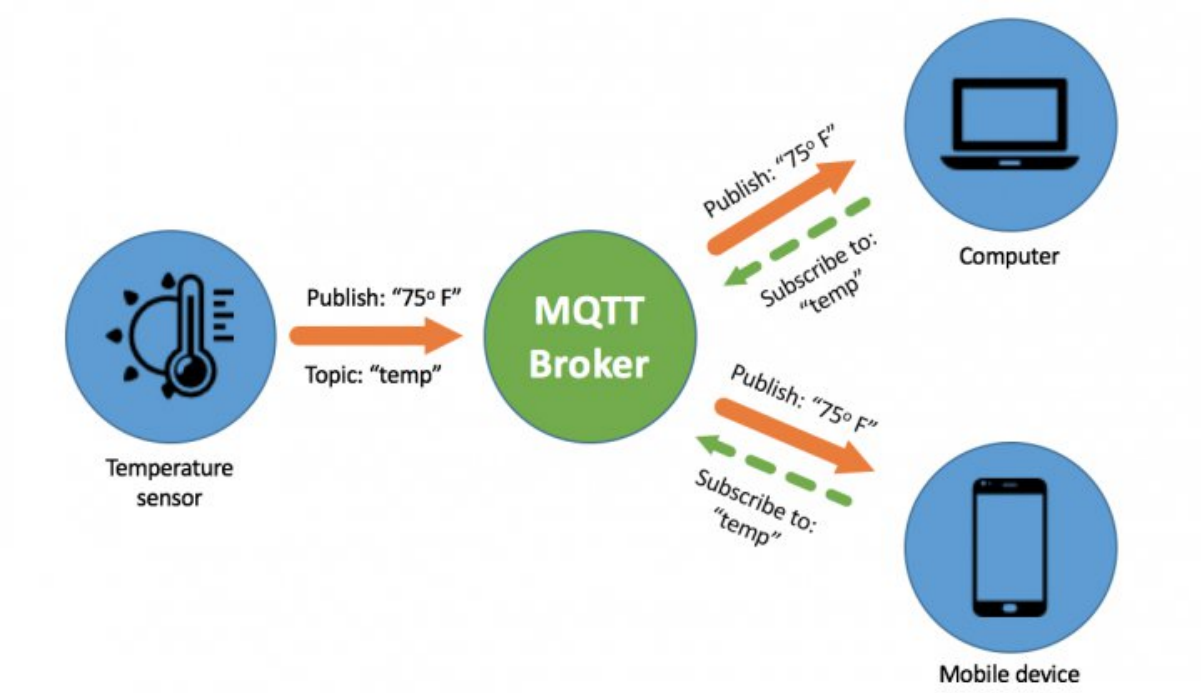
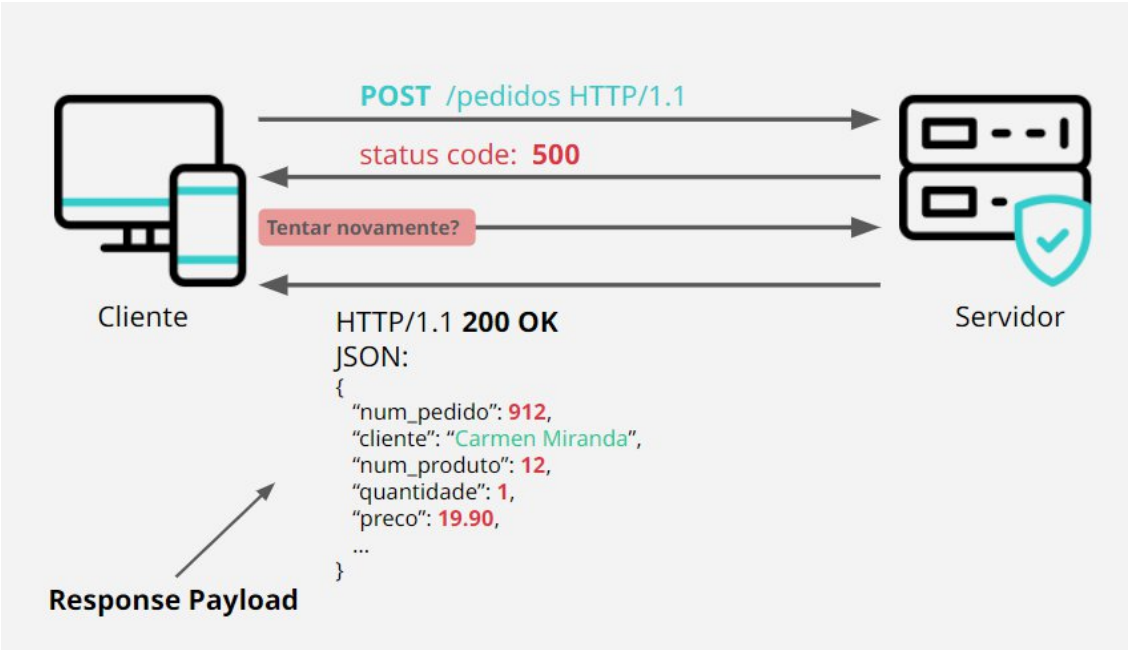


Figure 1.3: WebSockets vs. the traditional HTTP request/response model



Funcionamento de uma API

- Solicitação e Resposta: Como as APIs se comunicam?
  1. O cliente faz uma requisição HTTP para um *endpoint* da API.
  2. O servidor processa a solicitação e retorna uma resposta.
  3. A resposta pode conter dados, mensagens de erro ou códigos de status HTTP (200 OK, 404 Not Found, 500 Internal Server Error, etc.).



Funcionamento de uma API

- Exemplo Prático: API de um serviço de pagamento
  - Uma API como a do PayPal ou Stripe permite que um e-commerce processe pagamentos de clientes. O fluxo típico inclui:
    - Envio dos detalhes do pagamento;
    - Processamento e verificação da transação;
    - Retorno da resposta confirmando o pagamento ou indicando um erro;

Funcionamento de uma API

- Segurança
  - Autenticação: Tokens JWT, OAuth 2.0, API Keys;
  - Autorização: Controle de acesso baseado em funções (RBAC, ABAC).
  - Criptografia: Uso de HTTPS e certificados SSL para proteger os dados

Vantagens das APIs

- Modularidade:
  - APIs permitem que sistemas sejam desenvolvidos como componentes independentes, facilitando manutenção e escalabilidade.
- Reutilização de Código:
  - APIs possibilitam que funcionalidades sejam reaproveitadas em diferentes aplicações, reduzindo tempo e custo de desenvolvimento.
- Inovação:
  - Empresas podem criar novos serviços ao integrar APIs de terceiros, como Google Maps para geolocalização ou OpenAI para inteligência artificial.

Desenvolvimento de APIs

- Planejamento
  - Definição dos endpoints e recursos.
  - Escolha dos métodos HTTP adequados.
  - Estruturação dos dados e padrões de resposta.
- Implementação
  - Uso de frameworks como Express.js (Node.js), Flask (Python), Spring Boot (Java).
  - Escolha de banco de dados (SQL, NoSQL).
  - Documentação com Swagger ou OpenAPI.
- Testes
  - Testes unitários: Validação de funções individuais.
  - Testes de integração: Verificação da interação entre componentes.
  - Testes de aceitação: Confirmação de que a API atende aos requisitos do usuário.

Exemplos de APIs

- APIs Públicas
  - Google Maps API: Integração de mapas e geolocalização.
  - Twitter API: Acesso a dados de postagens e interações.
  - OpenWeather API: Consulta de previsão do tempo.
- APIs Privadas
  - APIs internas de empresas para comunicação entre sistemas internos.
  - APIs utilizadas para integração entre microserviços dentro de um ambiente corporativo.

Conclusão

- Resumo dos pontos principais
  - As APIs são essenciais para a conectividade e interoperabilidade dos sistemas modernos. Elas permitem modularidade, reutilização de código e inovação, além de viabilizarem soluções escaláveis e eficientes.
- Futuro das APIs
  - APIs baseadas em IA: Integração com modelos de aprendizado de máquina;
  - API-first Development: Estratégia onde APIs são o foco inicial do desenvolvimento de software;
  - Maior uso de GraphQL e gRPC: Para aumentar a flexibilidade e eficiência;
  - Automação e API Management: Ferramentas como Kong, Apigee e AWS API Gateway para controle e monitoramento de APIs.