VEM SER

Testes de API Rest com REST Assured


DBC

Aula 1 - Conceitos básicos para Testes de API Rest

Sumário

1. O que é uma API
2. Como funciona uma API Rest
3. Estrutura da Requisição
4. Estrutura da Resposta
5. Entendendo o JSON
6. Swagger
7. Heurística VADER
8. Kahoot
9. Task - RA01
10. Referências


O que é uma
API?

O que é uma API

- API é um acrônimo para *Application Programming Interface*, ou, em português, **Interface de Programação de Aplicação**.

- É um conjunto de rotinas e padrões estabelecidos por um software para a utilização das suas funções, dados e recursos por outro software, sem que este precise conhecer detalhes da sua implementação.

- Ou seja, a API permite expor ao mundo exterior as funções de um software sem que haja a necessidade de acesso ao seu código fonte.

O que é uma API

- As APIs podem ser implementadas de várias maneiras, contudo, o foco do módulo será sobre APIs Rest.

- **Mas o que é Rest?**
  - **RE**presentational **S**tate **T**ransfer (Transferência de Estado Representacional) é um estilo de arquitetura e fornece padrões de comunicação entre sistemas na web.
  - Por exemplo:
    - Arquitetura Client-server;
    - Stateless.

Saiba mais: https://www.toolsqa.com/rest-assured/what-is-rest/

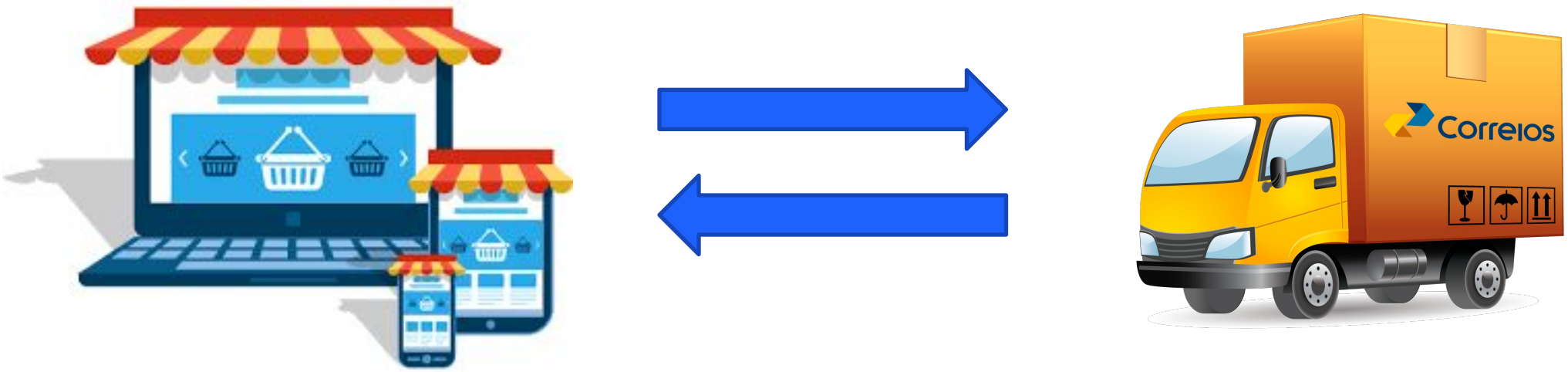
Como funciona uma API Rest

- A API funciona como um contrato, permitindo a comunicação entre dois sistemas. Essa comunicação consiste em duas partes: a requisição e a resposta.

Como funciona uma API Rest

- **Exemplo 1:** Podemos imaginar uma loja virtual onde um usuário deseja Buscar, Inserir, Alterar ou Excluir produtos pelo App.

Como funciona uma API Rest



- **Exemplo 2:** Ainda com o exemplo da loja virtual, vamos pensar que ela fornece seus produtos juntamente com o preço do frete. Para este valor ser fornecido é preciso que uma API se comunique com os Correios para validar o custo do envio do produto.


Estrutura
da
Requisição

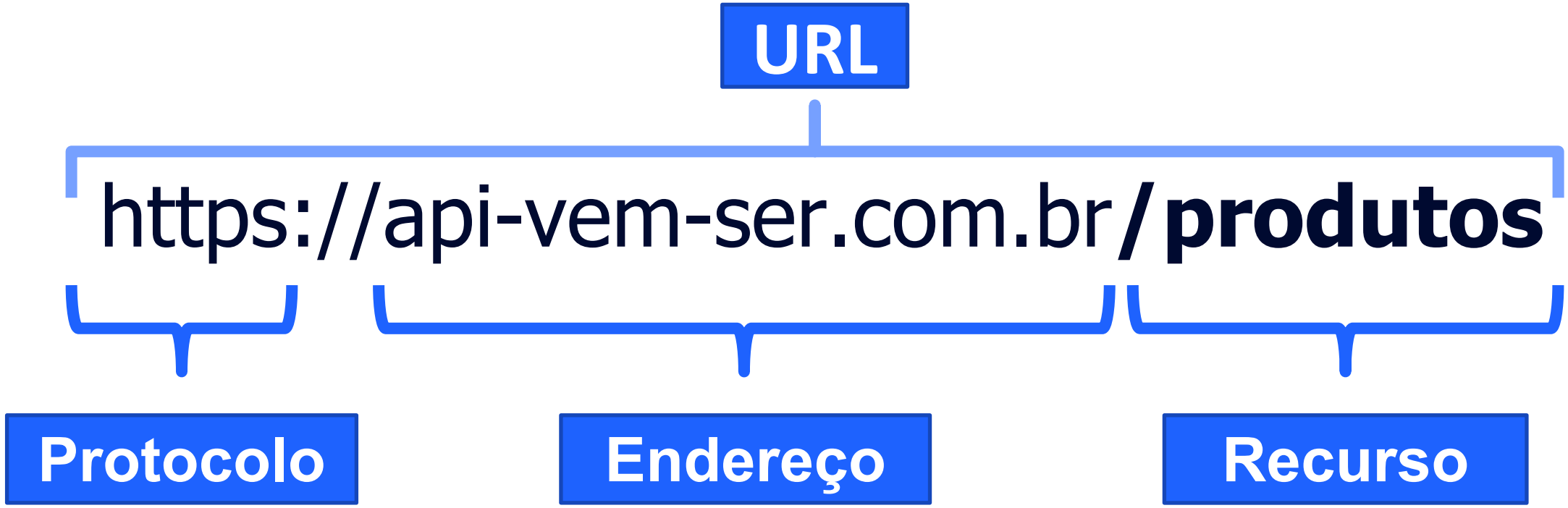
Estrutura da Requisição

- **Requisição ou Request:** É o pedido/solicitação que o cliente realiza ao servidor.

- **Estrutura:**
  - Endpoint / URL
  - Método
  - Cabeçalhos
  - Corpo

Endpoint / URL

- O termo URL significa Uniform Resource Locator, ou Localizador Uniforme de Recursos. É o endereço de um recurso informático disponível em uma rede.



Métodos HTTP

- Especificam a ação que deve ser executada por meio da solicitação. Também são conhecidos como verbos e geralmente utilizados para operações de CRUD (Create, Read, Update e Delete).

- Os mais utilizados são:
  - **GET:** Solicitar informações do servidor.
  - **POST:** Enviar dados para o servidor.
  - **PUT:** Atualiza dados no servidor.
  - **DELETE:** Exclui as representações de recursos do servidor.

Saiba mais: https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Methods

Cabeçalhos

- Cabeçalhos ou headers são metadados utilizados para enviar informações extras ao servidor sobre a solicitação.

- Exemplos:
  - Content type;
  - Authorization;
  - Cookies.

Corpo

- Trata-se dos dados que estamos enviando para o servidor por meio dos métodos **POST e PUT**, principalmente.

- Formatos: **JSON ou XML**.


Estrutura
da
Resposta

Estrutura da Resposta

- **Resposta ou Response:** É o resultado de uma solicitação enviada pelo cliente ao servidor. Essa resposta pode conter os dados que o cliente esperava receber ou uma resposta informando que algo deu errado.

- Formatos: **JSON e XML**

- **Estrutura:**
  - Status Code;
  - Cabeçalhos;
  - Corpo.

Status Code

- O código de status HTTP indica para o cliente qual a condição atual sobre o processamento de sua requisição, ou seja, se foi bem-sucedida ou não.

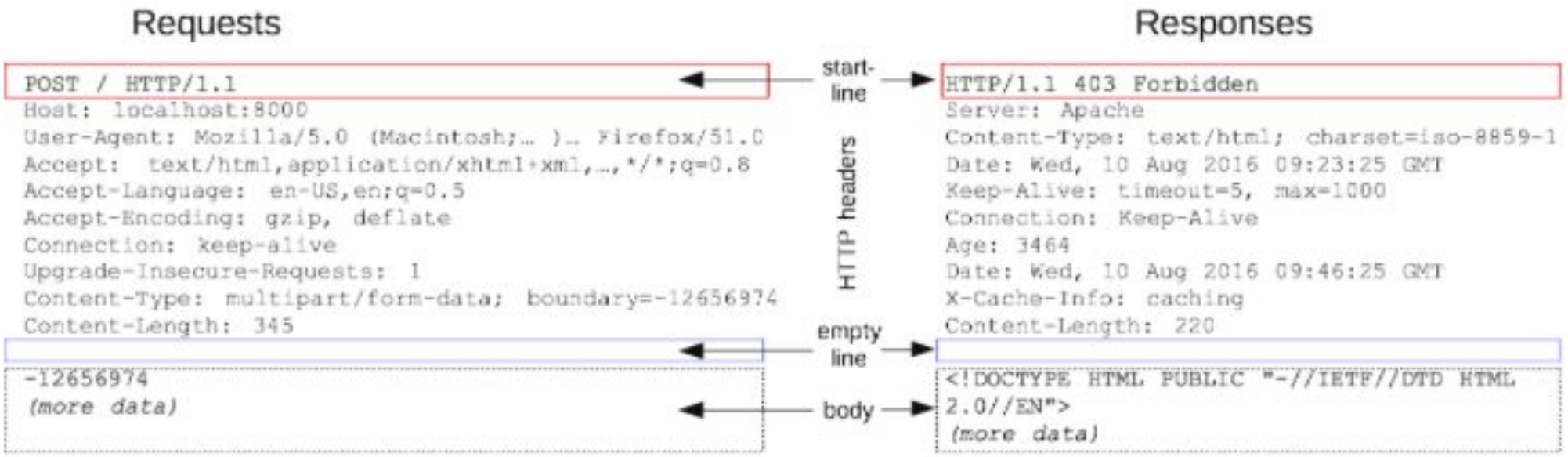
- As respostas são agrupadas em cinco classes:
  - 1xx - Informações
  - 2xx - Sucesso
  - 3xx - Redirecionamentos
  - 4xx - Erro do Cliente
  - 5xx - Erro do Servidor

Saiba mais: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Status Code

- **200 OK:** A solicitação foi bem sucedida.
- **201 Created:** A solicitação foi bem sucedida e um novo recurso foi criado.
- **400 Bad Request:** O servidor não vai processar a solicitação por um erro no corpo da requisição.
- **401 Unauthorized:** O cliente não forneceu as credenciais corretas para acessar o recurso.
- **403 Forbidden:** O servidor recebeu a requisição, mas se negou a autorizá-la.
- **404 Not Found:** O servidor não encontrou uma representação atual do recurso solicitado.
- **500 Internal Server Error:** O servidor encontrou uma condição inesperada que o impediu de atender completamente a requisição.

Requisição e Resposta




Entendendo
o
JSON

Entendendo o JSON

- **JSON** (Notação de Objetos JavaScript) é o formato de dados mais utilizado para o compartilhamento de informações entre sistemas.

```json
{
    "nome": "João da Silva",
    "idade": 20,
    "matricula": "2018123490",
    "curso": "Sistemas de Informação",
    "cadeiras": [
        "Estrutura de Dados",
        "Organização de Computadores",
        "Matemática Discreta"
    ]
}
```

Saiba mais: https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Objects/JSON

Intervalo

- 15 minutos
- Voltamos:


Swagger

Swagger

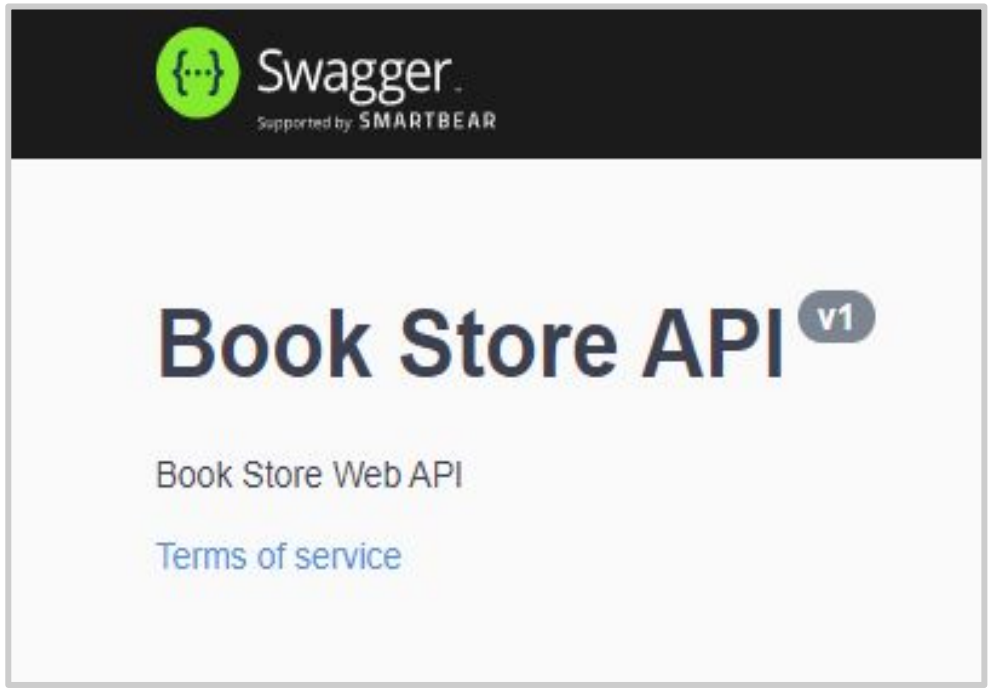
- Para testar uma API Rest, devemos entender a interface de uma API e isso é possível através da sua documentação. **O formato mais conhecido para se documentar uma API é o Swagger.**

- Com ele, temos acesso a:
  - Descrições dos recursos;
  - Métodos e seus endpoints;
  - Parâmetros utilizados;
  - Exemplos de Requisições e Respostas.

Swagger

- Para essa a aula, vamos utilizar a Book Store API, disponível em: https://bookstore.toolsqa.com/swagger/#/

- Para acompanhar: https://demoqa.com/books



Swagger

- **Para refletir**:

Durante a análise de um Swagger seria possível identificar bugs?


Heurística
VADER

Heurística VADER

- Heurística é um "método de investigação baseado na aproximação progressiva de um dado problema".

- A heurística VADER tem como objetivo ser um guia para avaliar o comportamento de uma API Rest durante testes exploratórios.

Verbs (Verbos);
Authentication and Authorization (Autenticação e Autorização);
Data (Dados);
Errors (Erros);
Responsiveness (Tempo de resposta).

Heurística VADER

- **Verbos:** Consiste em testar os métodos aptos e não aptos para o endpoint.
- **Autenticação e Autorização:** Validar token, testes com token ou usuário e senha inválido ou inexistentes, restrições de acesso assim que for autorizado, etc.
- **Dados:** Serialização, tipagem, paginação, formato (JSON / XML) e tamanho.
- **Erros:** Status code das respostas, mensagens.
- **Tempo de resposta:** Analisar o tempo de resposta e criar um padrão para o projeto.


Kahoot

Hora do jogo!

Valendo um Chevette SL 1990

Kahoot!


Task RA-01

TASK RA-01

- Utilizando a heurística VADER, realizar testes exploratórios através do Swagger da Book Store API.

- Deve ser documentado seguindo o seguinte padrão:
  - “A - Enviei um token inválido e retornou 500 ao invés de 401”.
  - “D - Enviei o userId inexistente na criação da lista de livros, retornou mensagem de ‘User Id not correct!’, mas status 401 Unauthorized.

- A entrega deve ser realizada no Classroom.

Referências

Referências

- https://www.toolsqa.com/rest-assured/what-is-rest/
- https://www.toolsqa.com/client-server/http-request/
- https://www.toolsqa.com/client-server/http-response/
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Status
- https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Objects/JSON
- https://www.toolsqa.com/rest-assured/api-documentation/
- https://maximilianoalves.medium.com/vader-heuristica-para-teste-de-api-na-pratica-fcf78c6acec
- https://qa-matters.com/2016/07/30/vader-a-rest-api-test-heuristic/


Let's Tech Up Together