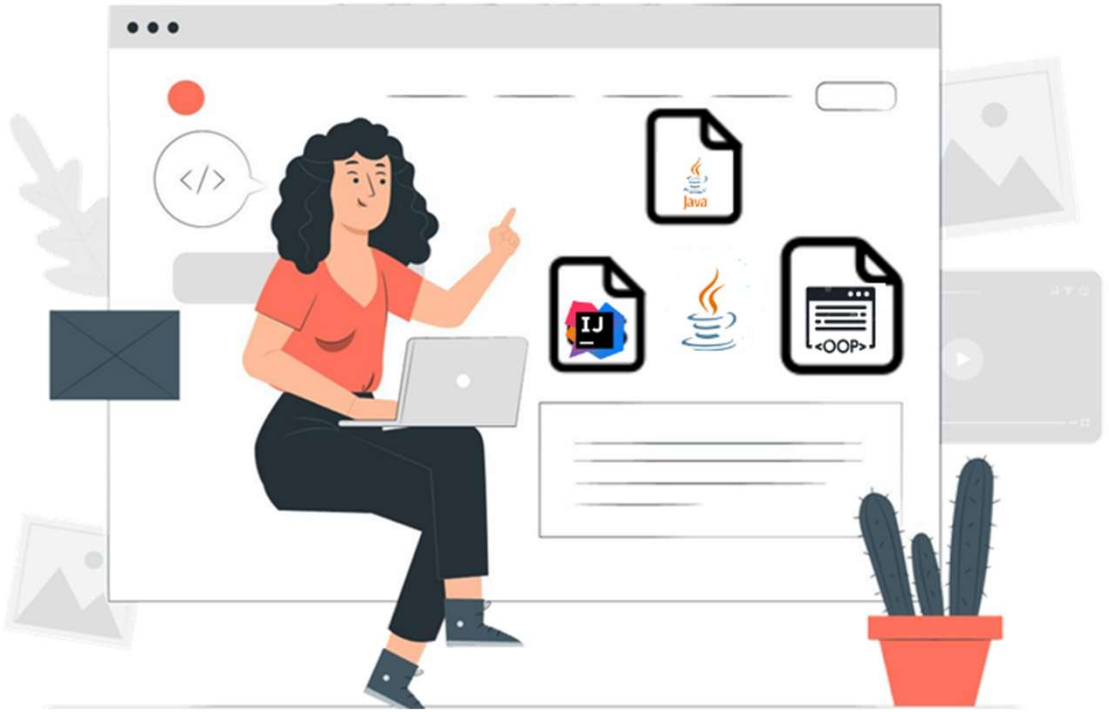


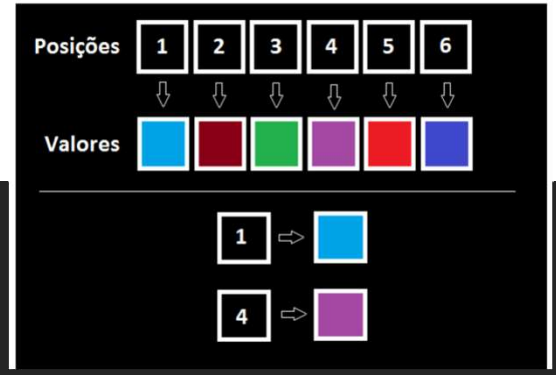
Domain Driven Design

Eliane Marion

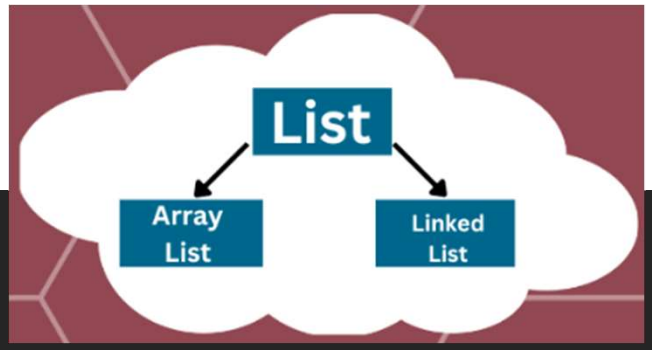
FIAP

2024

AGENDA DE TRABALHO



VETORES

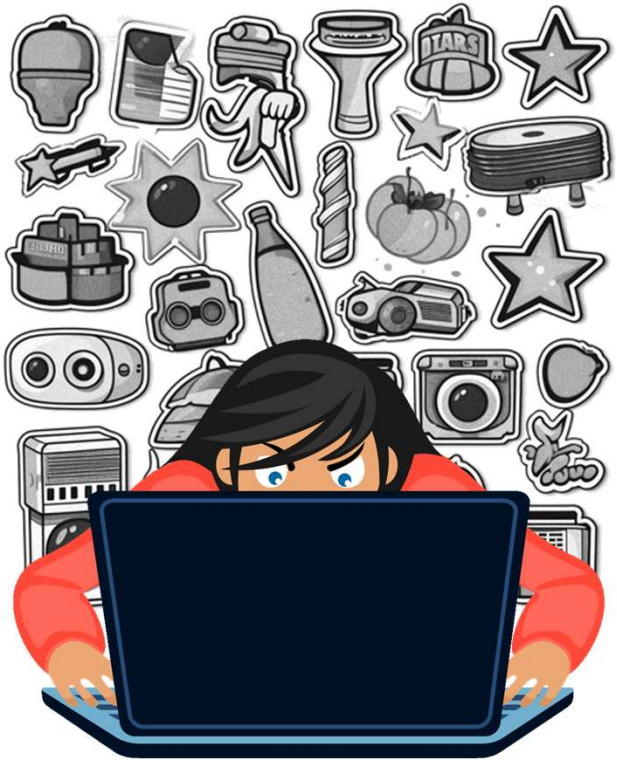


LISTAS

01

JSON

JSON



- O formato JSON (JavaScript Object Notation) é, como o nome sugere, uma forma de notação de objetos JavaScript, de modo que eles possam ser representados de uma forma comum a diversas linguagens.
- Facilmente trafegado entre aplicações em quaisquer protocolos, inclusive o HTTP.
- É um texto.

REPRESENTAÇÃO

{}

```
{
    "nome": "tênis",
    "descricao": "tênis de corrida",
    "quantidadeEstoque": 15,
    "preco": 300
}
```

VANTAGENS

CHECK LIST

Vantagens do JSON

- Leitura mais simples
- É tipado
- Arquivo com tamanho reduzido
- Velocidade maior na execução e transporte de dados
- Quem utiliza? Google, Facebook, Yahoo!, Twitter...

JSON

JSON – é um formato de dados leve e de fácil leitura utilizado para troca de informações entre sistemas computacionais.

Amplamente utilizado na web para representar dados estruturados.

- Oferece simplicidade, legibilidade, portabilidade e suporte amplo.
- JSON é muito utilizado na comunicação entre servidores e clientes em aplicações web, inclusive em APIs (Interface de Programação de Aplicativos), para transferir dados entre servidor e clientes de forma mais eficiente.

Manipulando arquivo JSON

Primeiro, adicione a dependência Gson ao seu pom.xml se você estiver usando Maven:

{}

```
<dependency>
        <groupId>com.google.code.gson</groupId>
        <artifactId>gson</artifactId>
        <version>2.9.0</version>
</dependency>
```

ESCRITA DE JSON EM ARQUIVO

Usamos a biblioteca **Gson** para converter objetos Java em JSON e escrevê-los em arquivos. **setPrettyPrinting:** Gera JSON formatado de maneira legível.

```
public static void main(String[] args) {
    Pessoa pessoa = new Pessoa("João", 30);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();

    try (FileWriter writer = new FileWriter("pessoa.json")) {
        gson.toJson(pessoa, writer);
        System.out.println("Arquivo JSON escrito com sucesso!");
    } catch (IOException e) {
        System.out.println("Ocorreu um erro ao escrever o arquivo
JSON: " + e.getMessage());
    }
}
```

Gson: Facilita a conversão de objetos Java para JSON.

{}

LEITURA DE JSON DE UM ARQUIVO

É possível ler um arquivo JSON e convertê-lo em um objeto Java.

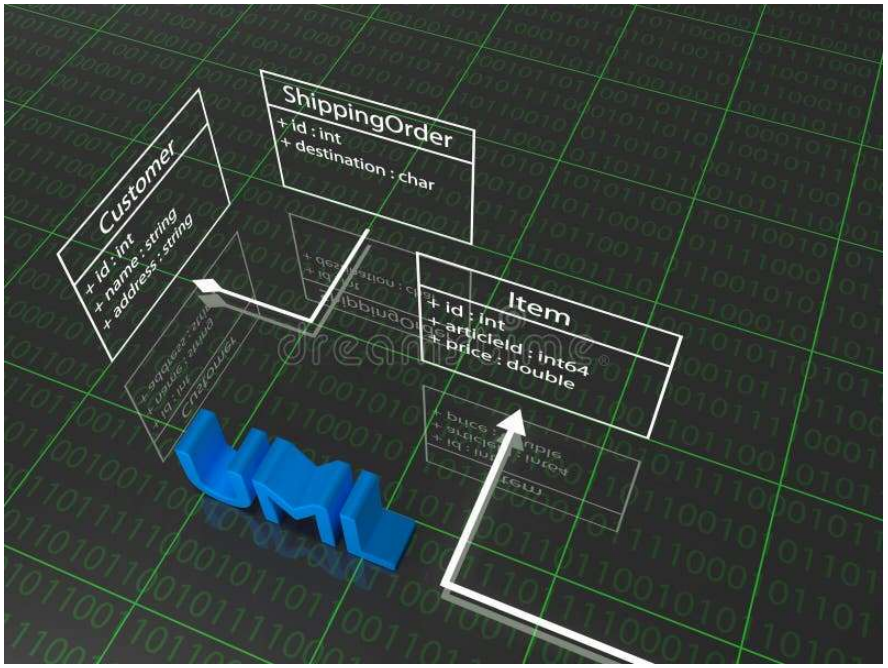
```
public static void main(String[] args) {
    Gson gson = new Gson();

    try (FileReader reader = new FileReader("pessoa.json")) {
        Pessoa pessoa = gson.fromJson(reader, Pessoa.class);

        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Idade: " + pessoa.getIdade());
    } catch (IOException e) {
        System.out.println("Ocorreu um erro ao ler o arquivo
JSON: " + e.getMessage());
    }
}
```

fromJson: Converte Json em objeto Java

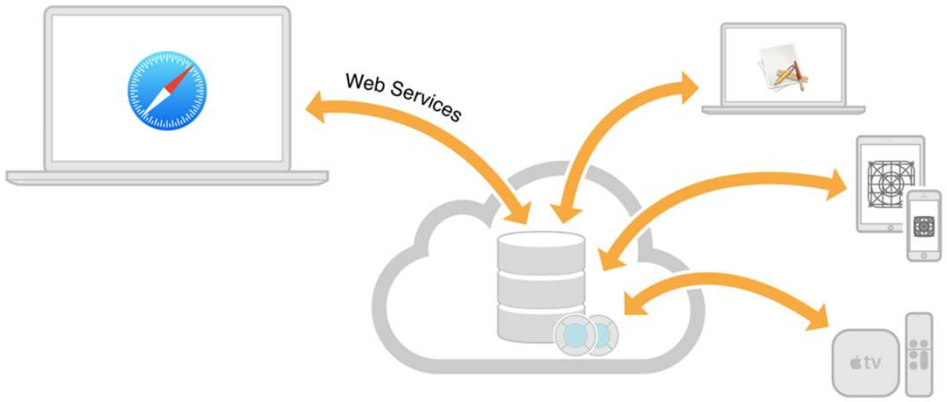
{}



02

WEB SERVICES

WEB SERVICES

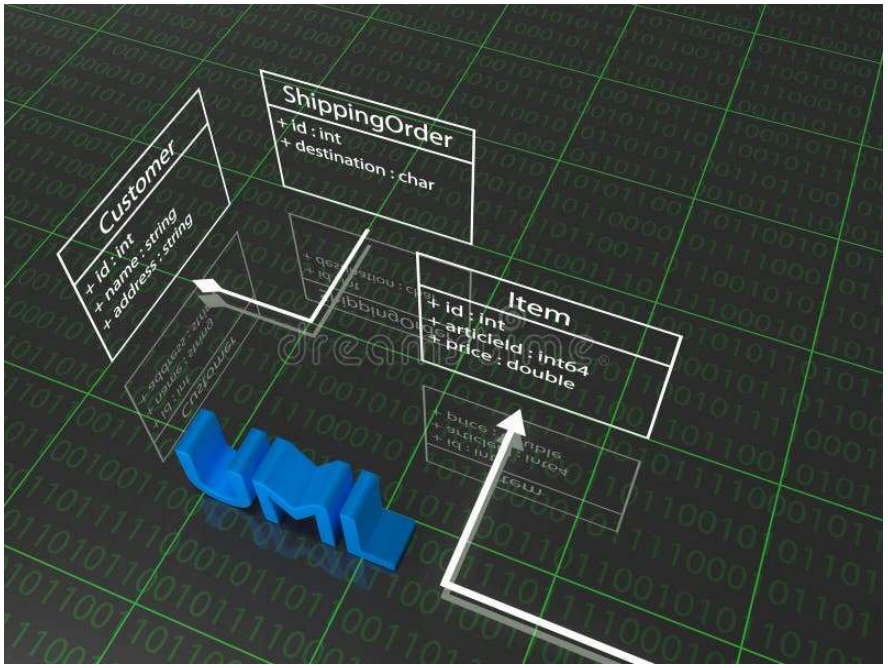


Integração e comunicação entre sistemas diferentes;

Independente de plataforma (Java, .NET, PHP, Ruby, etc..)

Permite o envio e recebimento de dados em formatos XML, Json, CSV, etc..

Os dois tipos mais comuns de web services: **SOAP** e **REST**;


ShippingOrder
+ id : int
+ destination : char
Customer
+ id : int
+ name : string
+ address : string
Item
+ id : int
+ articleId : int64
+ price : double
UML

03

REST API

REST

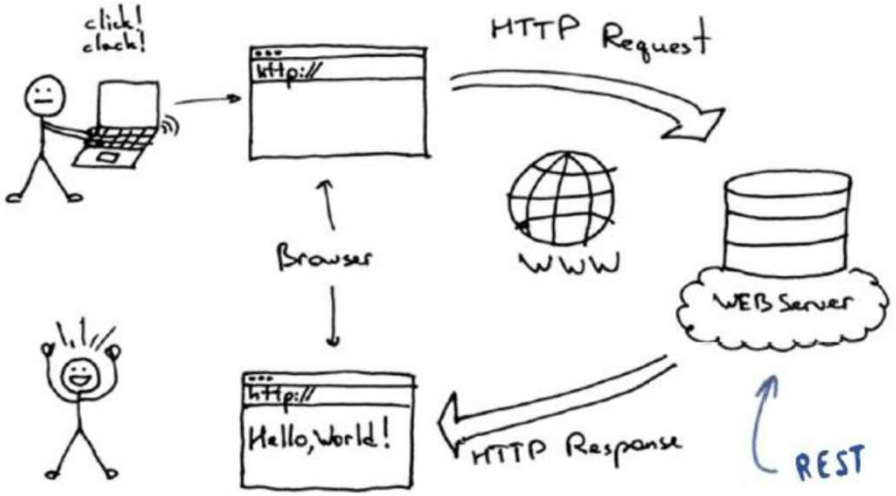
REST: define um conjunto de regras e boas práticas para o desenvolvimento de APIs que possibilitam a execução de solicitações e o recebimento de respostas via protocolo HTTP, como GET e POST, permitindo a comunicação entre aplicações.

RESTFul: Existe uma confusão entre os termos, entretanto a diferença é apenas gramatical. Sistemas que utilizam os princípios Rest são chamado **RESTFul**.

RESTFul (REpresentational State Transfer)

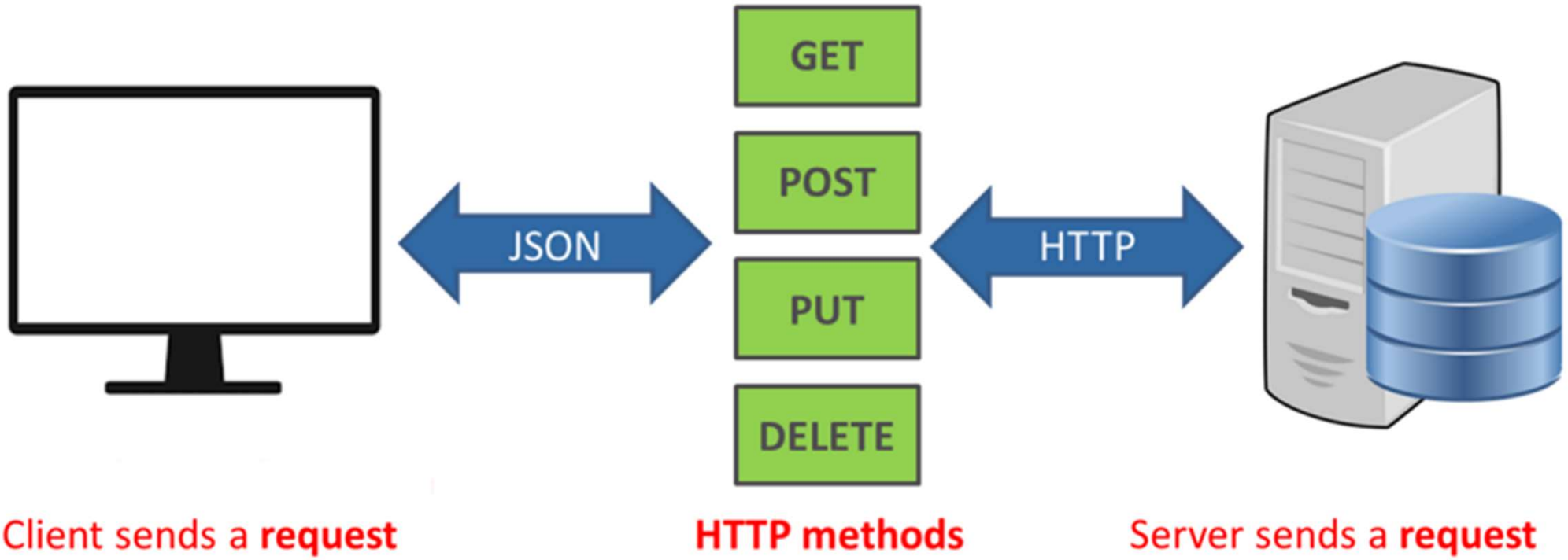
- Simples, leve, fácil de desenvolver e evoluir;
- Tudo é um recurso (Resource);
- Cada recurso possui um identificador (URI);
- Recursos podem utilizar vários formatos: html, xml, Json;
- Utiliza o Protocolo HTTP;
- Os métodos HTTP: GET, POST, PUT, DELETE

REST - Funcionamento



Toda a comunicação da interface REST é feita via web, ou seja, através de uma requisição (pedido feito pelo cliente) a uma URI (Uniform Resource Identifier), que referência um recurso, utilizando um dos quatro métodos HTTP (GET, PUT, POST ou DELETE) que, por sua vez, traz uma resposta.

REST


GET
POST
PUT
DELETE
JSON
HTTP
Client sends a **request**
HTTP methods
Server sends a **request**

URI – Unified Resource Identifier

Quando realizamos uma requisição, é preciso determinar o **endereço** do **recurso** que vamos acessar.

VERBO	URI	AÇÃO
POST	/contato/	Cria um novo recurso
GET	/contato/1	Visualizar / Recupera informações de um recurso
PUT	/contato/1	Alterar / Atualiza um recurso
DELETE	/contato/1	Apagar / Remove um recurso