**Nome: Lucas Rodrigues Da Silva**

OBS: (1) inserir no diário de bordo;

(2) Não esquecer de inserir as referências.

Faça uma pesquisa sobre REST, RESTFULL e JSON.

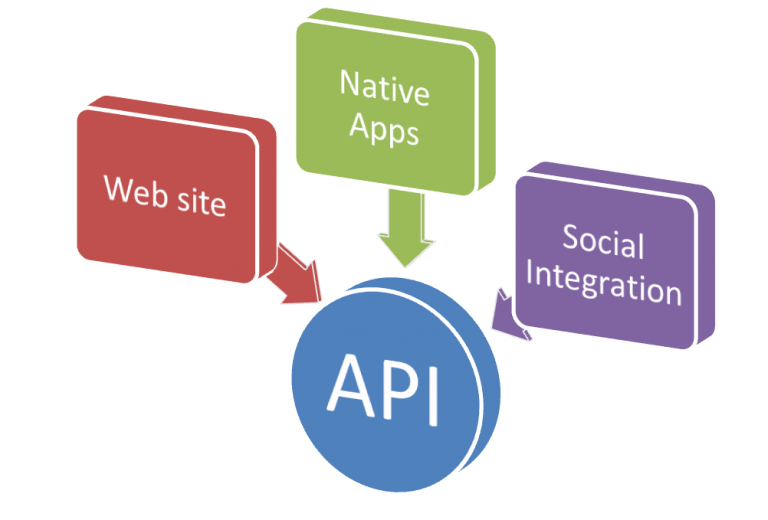
Aprofunde o máximo que puder com exemplos práticos em PHP e JS.

Compacte os arquivos de exemplo juntamente com o arquivo da avaliação e poste em seu diário de bordo.

# API

O acrônimo API que provém do inglês Application Programming Interface (Em português, significa Interface de Programação de Aplicações), trata-se de um conjunto de rotinas e padrões estabelecidos e documentados por uma aplicação A, para que outras aplicações consigam utilizar as funcionalidades desta aplicação A, sem precisar conhecer detalhes da implementação do software.

Desta forma, entendemos que as APIs permitem uma interoperabilidade entre aplicações. Em outras palavras, a comunicação entre aplicações e entre os usuários.



# REST

O HTTP é o principal protocolo de comunicação para sistemas Web, existente há mais de 20 anos, e em todo esse tempo sofreu algumas atualizações. Nos anos 2000, um dos principais autores do protocolo HTTP, Roy Fielding, sugeriu, dentre outras coisas, o uso de novos métodos HTTP. Estes métodos visavam resolver problemas relacionados a semântica quando requisições HTTP eram feitas.

REST é acrônimo de Representational State Transfer, e tem como objetivo primário a definição de características fundamentais para a construção de aplicações Web seguindo boas práticas.

A Web como a conhecemos hoje, funciona seguindo práticas REST, e para entendermos melhor, vamos considerar o exemplo abaixo:

1. Você entra com um endereço em seu navegador (Chrome, Firefox, Edge, etc).

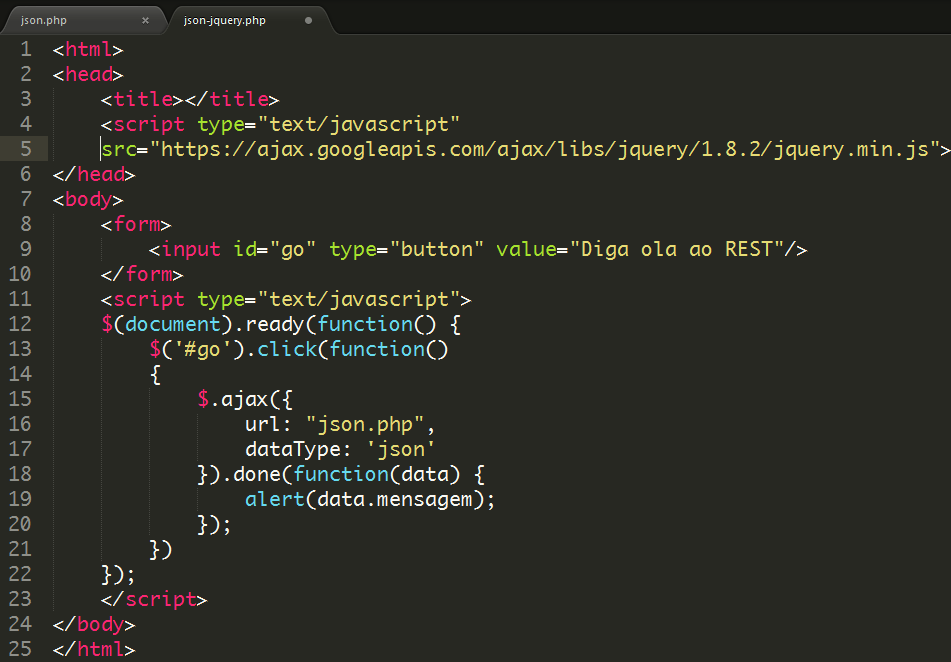
2. Seu navegador estabelece uma conexão TCP/IP com o servidor de destino e envia uma requisição GET HTTP com o endereço que você digitou.

3. O servidor interpreta sua requisição e de acordo com o que foi solicitado, uma resposta HTTP é retornada ao seu navegador.

4. A resposta retornada pode ser de sucesso, contendo alguma representação em formato HTML, ou pode ser algum erro, como por exemplo o famoso 404 Not Found, que indica que o endereço/recurso que você solicitou não pôde ser encontrado.

5. Em caso de sucesso, o seu navegador interpreta o HTML e você consegue navegar pela página renderizada.

Todo esse processo é repetido enquanto você está navegando em alguma página Web. Cada link que você clica ou formulário que submete, efetua os passos que discutimos acima.



Fonte : <https://www.youtube.com/watch?v=weQ8ssA6iBU>

# RESTFULL

API RESTful é uma interface que fornece dados em um formato padronizado baseado em requisições HTTP.

Por exemplo: A API do Facebook, que permite que você se autentique em aplicações externas ao Facebook (como o login da PlayStation Network, que é requisitada aos jogadores do PlayStation 4). Ela fornece dados do Facebook para essas aplicações, facilitando o cadastro e o acesso.

API RESTful fica parada até que acontece uma requisição. É como um carro estacionado que só é ativado após a ignição com a chave. No caso do exemplo do login da PSN usando os dados do Facebook, a API do Facebook não fica processando os dados de todos os seus usuários continuamente, ela espera até que um usuário peça a permissão para se autenticar.

As APIs Restful aumentam a performance para situações de concorrência, ou seja, quando muitas pessoas estão pedindo a mesma coisa ao mesmo tempo. Elas utilizam verbos para definir qual é a finalidade da requisição que está sendo enviada. Os verbos são:

GET: A requisição é um pedido de dados para a API. A API vai buscar os dados solicitados em algum banco e, provavelmente, vai retornar em formato JSON (formato de notação de objeto JavaScript);

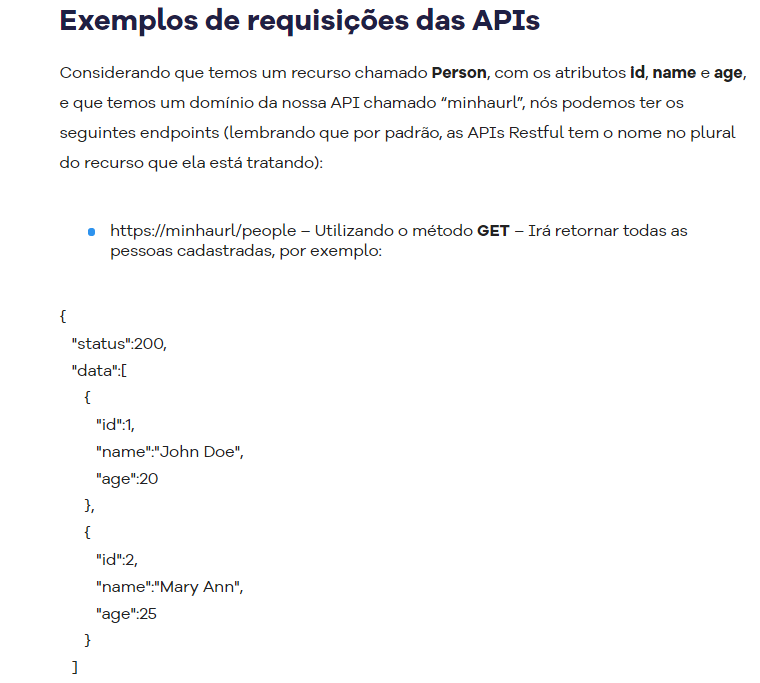
POST: Tipo de requisição utilizada para criar um recurso em uma determinada API. São chamados de recursos o objeto que está sendo tratado naquela API.

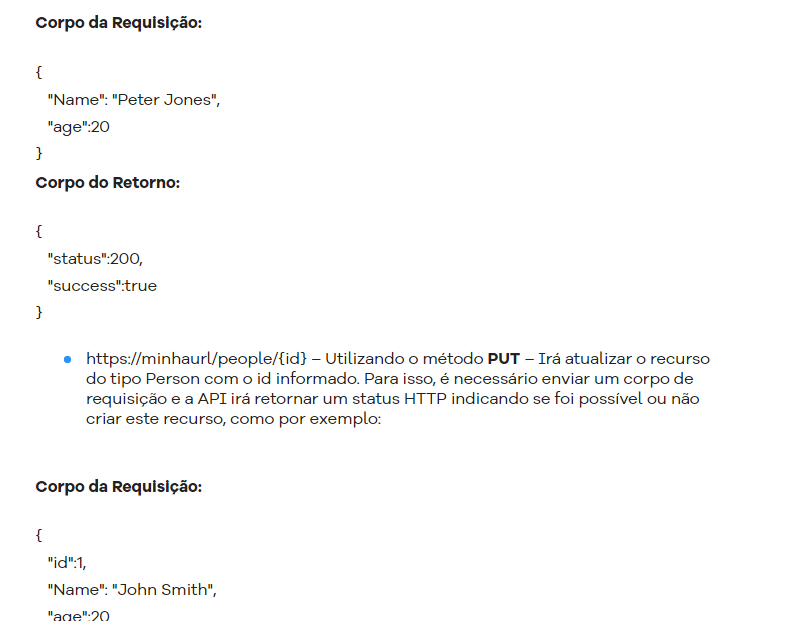
PUT: Requisição utilizada para atualizar o recurso indicado com alguma informação.

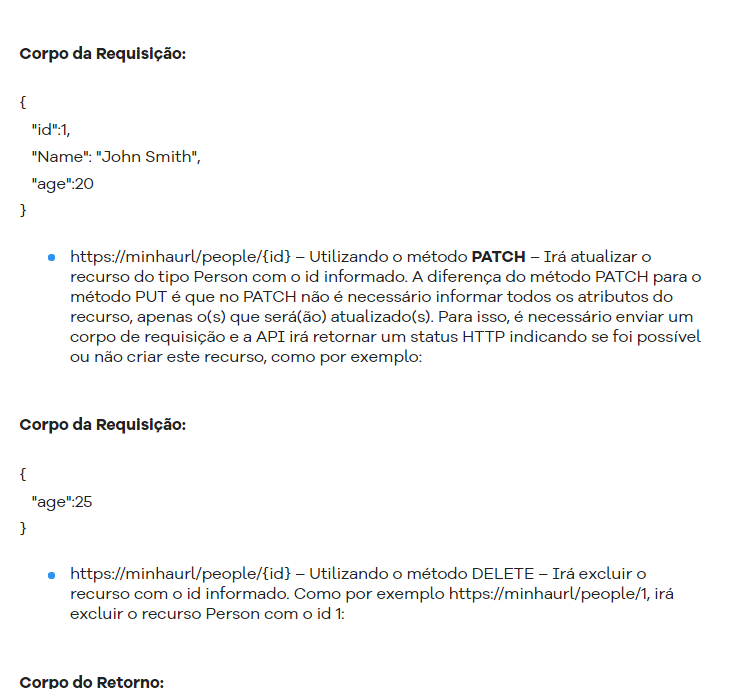
PATCH: Requisição feita para atualização de somente uma parte de um recurso.

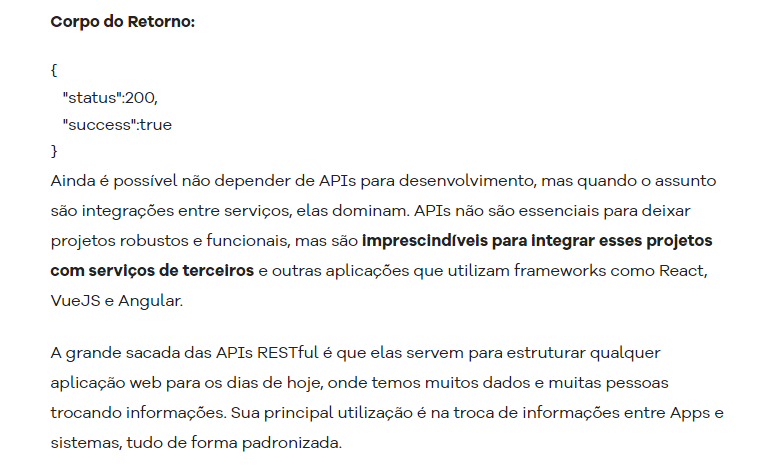
DELETE: Requisição para excluir um dado.

VIDEO : <https://www.youtube.com/watch?v=CDQH4CXZgVY>









# JSON

JSON é basicamente um formato leve de troca de informações/dados entre sistemas. Mas JSON significa JavaScript Object Notation, ou seja, só posso usar com JavaScript correto? Na verdade não e alguns ainda caem nesta armadilha.

O JSON além de ser um formato leve para troca de dados é também muito simples de ler. Mas quando dizemos que algo é simples, é interessante compará-lo com algo mais complexo para entendermos tal simplicidade não é? Neste caso podemos comparar o JSON com o formato XML.



Vantagens do JSON:

Leitura mais simples

Analisador(parsing) mais fácil

JSON suporta objetos! Sim, ele é tipado!

Velocidade maior na execução e transporte de dados

Arquivo com tamanho reduzido

Quem utiliza? Google, Facebook, Yahoo!, Twitter...

Estas são algumas das vantagens apresentadas pelo JSON. Agora vamos ao que interessa: Código! Vamos fazer um exemplo extremamente simples nesta primeira parte e avançaremos no próximo artigo, inclusive falando sobre JSON em páginas Web.

Qual biblioteca usar?

Existem diversas bibliotecas para trabalharmos com JSON e Java. Usaremos no nosso estudo o json.jar que você pode baixar tranquilamente neste link.