Insper

Programação Eficaz

REST e APIs

O que é uma API?

API

Application Programming Interface

Interação entre aplicações ou dentro da própria aplicação através de programação.

Ex: uma aplicação fazendo um post automaticamente na sua rede social favorita

1 United States Dollar equals

5.08 Brazilian Real

Jun 3, 22:33 UTC · Disclaimer

1	United States Dollar
5,08	Brazilian Real

Data provided by Morningstar for Currency and Coinbase for Cryptocurrency

E-commerce (mais exemplos)

- Quais serviços web um e-commerce precisa acessar?

E-commerce (mais exemplos)

- Quais serviços web um e-commerce precisa acessar?
  - Valor do frete
  - Transação do cartão de crédito
  - Login por redes sociais

Principais tipos de APIs

- **APIs abertas:** (ou públicas) sem restrições de acesso;
- **APIs de parceiros:** é necessário algum direito/licença para acessar;
- **APIs internas:** (ou privadas) interação entre produtos/serviços internos de uma mesma empresa;

O tal do REST

REST - Representational State Transfer

Pode ser considerado como um **conjunto de princípios**, que quando aplicados de maneira correta em uma aplicação, a beneficia com a arquitetura e padrões da própria Web.

Um sistema que adere a esses princípios é chamado de **RESTFul**.

Foi descrito por Roy Fielding, um dos principais criadores do protocolo HTTP, em sua tese de doutorado e que foi adotado como o modelo a ser utilizado na evolução da arquitetura do protocolo HTTP.

REST - Representational State Transfer

Obter (GET)

Criar (POST)

Substituir (PUT)

RECURSO

Atualizar parcialmente (PATCH)

Excluir (DELETE)

Recursos

Um recurso é um elemento abstrato e que nos permite mapear qualquer coisa do mundo real como um elemento para acesso via Web.

"The key abstraction of information in REST is a resource. Any information that can be named can be a resource: a document or image and so on.." - Roy Fielding

Identificação de recursos

URI

Uniform
Resource
Identifier

http://servicorest.com.br/produtos

http://servicorest.com.br/clientes

http://servicorest.com.br/clientes/57

http://servicorest.com.br/vendas

CRUD

O CRUD vem do inglês, das palavras (Create, Read, Update, Delete) que quer dizer basicamente as 4 (quatro) principais operações com um banco de dados (inserir, ler, atualizar, excluir).

Como é muito comum que web services manipulem dados em uma base a associação entre os verbos e as operações do CRUD é muito comum.

Acessando e alterando recursos

Método	URI	Utilização
GET	/clientes	Recuperar os dados de todos os clientes.
GET	/clientes/id	Recuperar os dados de um determinado cliente.
POST	/clientes	Criar um novo cliente.
PUT	/clientes/id	Atualizar os dados de um determinado cliente
DELETE	/clientes/id	Excluir um determinado cliente.

Tipos de dados

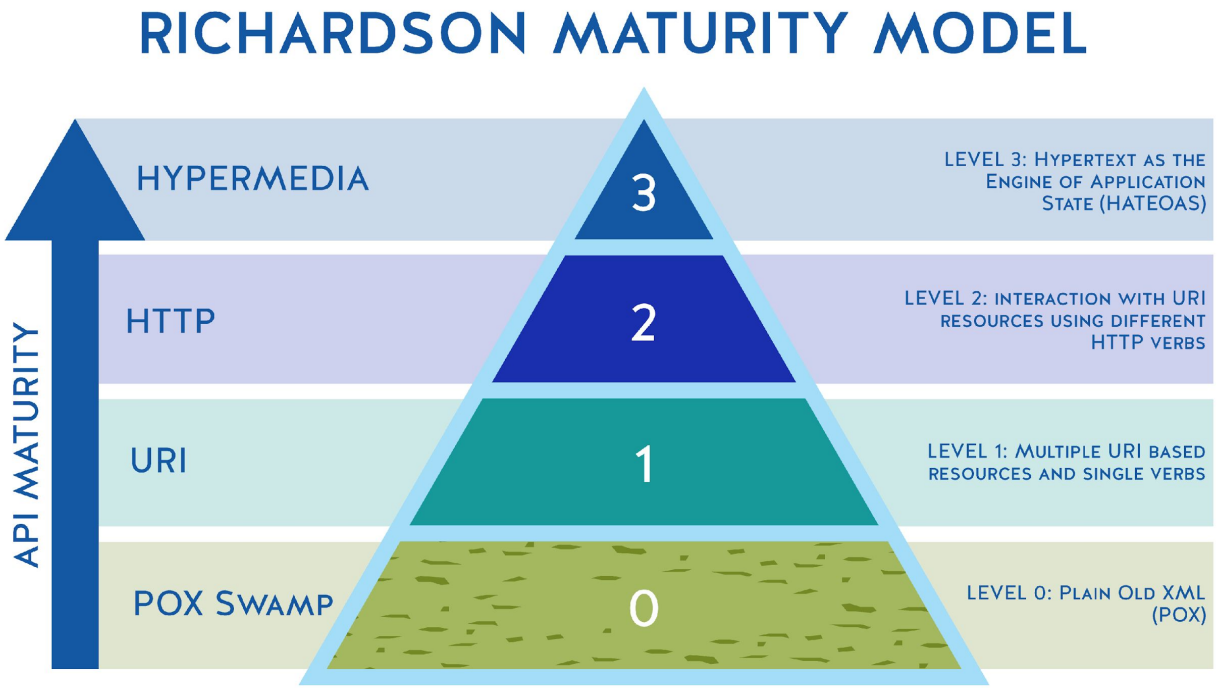
XML

```
<employee>
  <id>1</id>
  <name>John Doe</name>
</employee>
```

JSON

```
{
  "employee": {
    "id": 1,
    "name": "John Doe"
  }
}
```

Modelo de Maturidade de Richardson



https://martinfowler.com/articles/richardsonMaturityModel.html

Nível 0 - POX

- Nenhum padrão
- Recursos mal definidos

RPC (POX)		
Verbo HTTP	**URI**	**Ação**
GET	/buscarCliente/1	Visualizar
POST	/salvarCliente	Criar
POST	/alterarCliente/1	Alterar
GET/POST	/deletarCliente/1	Remover

Nível 1 - Recursos

- Uso eficiente das URIs e recursos mapeados corretamente
- Verbos mal definidos (praticamente só GET e POST)

REST		
Verbo HTTP	**URI**	**Ação**
GET	/cliente/1	Visualizar
POST	/cliente	Criar
POST	/cliente/1	Alterar
POST	/cliente/1	Remover

Nível 2 - Uso correto

- Uso eficiente das URIs e verbos HTTP

REST		
Verbo HTTP	**URI**	**Ação**
GET	/cliente/1	Visualizar
POST	/cliente	Criar
PUT	/cliente/1	Alterar
DELETE	/cliente/1	Remover

Nível 3 - HATEOAS (Hypermedia as the Engine of Application State)

- O objetivo dos controles hipermídia é que eles nos digam o que podemos fazer a seguir e o URI do recurso que precisamos manipular para fazê-lo.

```
GET /cliente/1

HTTP/1.1 200 OK

<Cliente>

    <Id>1</Id>

    <Nome>Manoel da Silva</Nome>

    <link rel="deletar" href="/cliente/1" />

    <link rel="notificar" href="/cliente/1/notificacao" />

</Cliente>
```

```
{
 "Cliente": {
  "Id": 1,
  "Nome": "Manoel da Silva",
  "deletar": "/cliente/1",
  "notificar": "/cliente/1/notificacao"
 }
}
```

Falando em URIs...

Utilize nomes legíveis (por humanos)

Isso facilita a vida dos clientes que utilizarão o serviço, além de reduzir a necessidade de documentações extensas.

Utilize o mesmo padrão de URI na identificação dos recursos

- http://servicorest.com.br/produto (Singular);
- http://servicorest.com.br/clientes (Plural);
- http://servicorest.com.br/processosAdministrativos (Camel Case);
- http://servicorest.com.br/processos_judidicais (Snake Case).

Evite adicionar na URI a operação a ser realizada no recurso

- http://servicorest.com.br/produtos/cadastrar;
- http://servicorest.com.br/clientes/10/excluir;
- http://servicorest.com.br/vendas/34/atualizar.

Falando em URIs...

Evite adicionar na URI o formato desejado na representação do recurso

- http://servicorest.com.br/produtos/xml;
- http://servicorest.com.br/clientes/112?formato=json.

Evite alterações nas URI's

A URI é a porta de entrada de um serviço. Se você a altera, isso certamente causará impacto nos clientes que estavam a utilizando, pois você alterou a forma de acesso a ele. Após definir uma URI e disponibilizar a manipulação de um recurso por ela, evite ao máximo sua alteração.

Nos casos mais críticos, no qual realmente uma URI precisará ser alterada, notifique os clientes desse serviço previamente. Verifique também a possibilidade de se manter a URI antiga, fazendo um redirecionamento para a nova URI.

Utilize os códigos HTTP corretamente

Fonte: https://httpstatusdogs.com/

Códigos HTTP

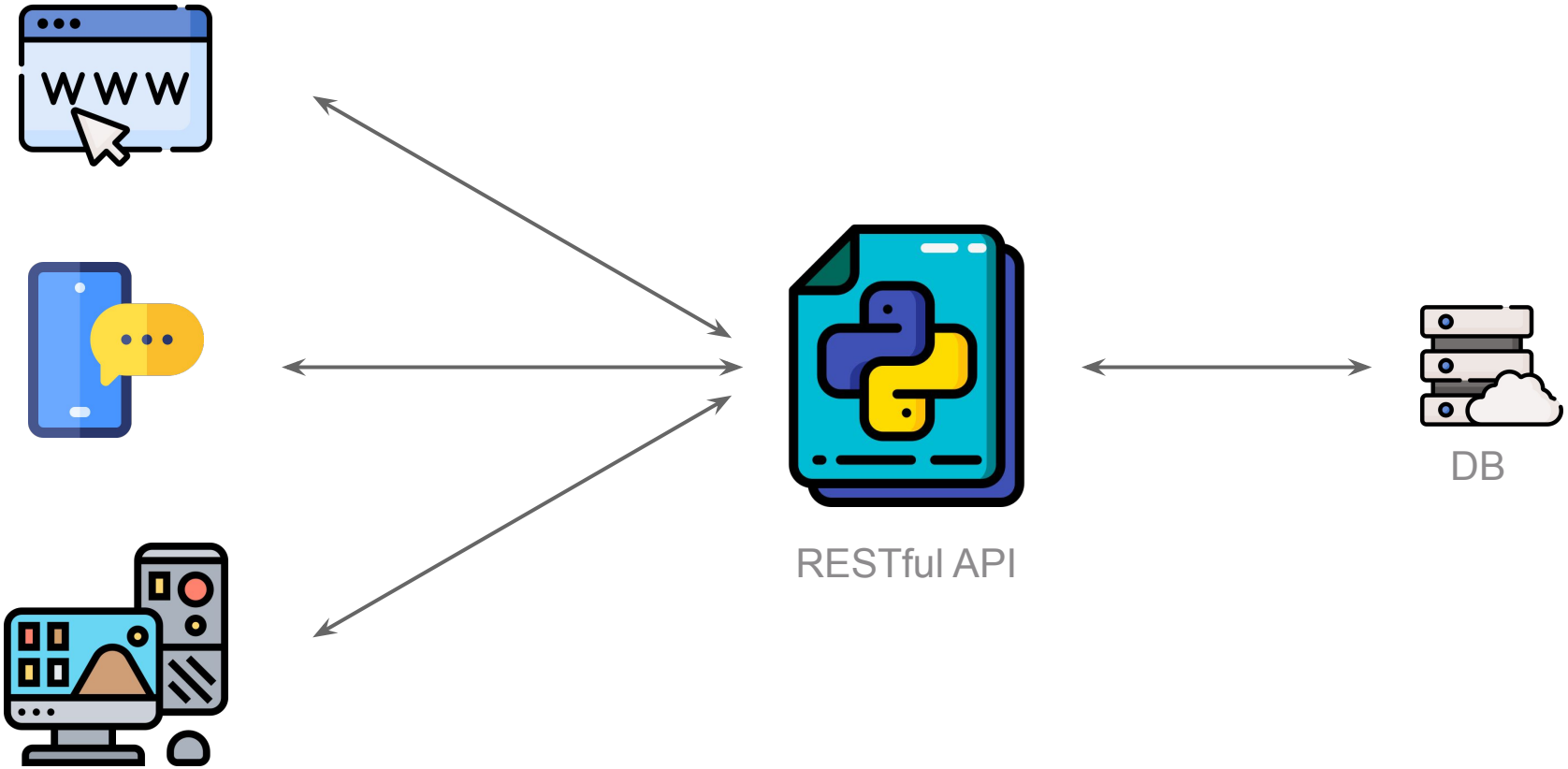
O servidor deve processar cada uma das requisições e retornar uma resposta adequada.

- 1XX – Informações Gerais
- 2XX – Sucesso
- 3XX – Redirecionamento
- 4XX – Erro no cliente
- 5XX – Erro no servidor

Resumindo...

O que é REST

- Maneira de projetar um serviço web
- Baseado principalmente nos 4 verbos HTTP
  - GET - Requisitar informação
  - POST - Enviar informação
  - PUT - Armazenar informação
  - DELETE - Excluir informação
- Orientado a recursos


WWW
RESTful API
DB

Exemplos de APIs

- LoL: https://developer.riotgames.com/apis#account-v1

- Google Maps: https://developers.google.com/maps/documentation/urls/get-started

- Twitter: https://developer.twitter.com/en/docs/twitter-api

Até a próxima aula!

Márcio F. Stabile jr.
marciofsj@insper.edu.br