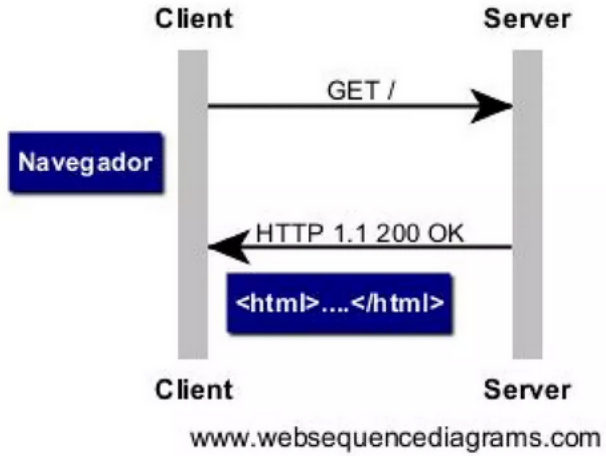
APIs REST

Spring REST / Spring Data

HTTP

HTTP Características

- Método Request - Response
- URL (Uniform Resource Locator)
- Verbos GET, POST, PUT, etc..
- Permite Negociação de Dados
- StateLess



HTTP Requests

Verbo (GET, POST, PUT, PATCH, DELETE)

URI (Uniform Resource Identifier)

GET /users/douglas HTTP/1.1

```
Host: example.net
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:44.0) Gecko/20100101 Firefox/44.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

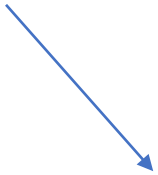
Header

HTTP

Http Responses

HTTP/1.1 **200 OK**
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
....



Body

Status Code “Comuns”

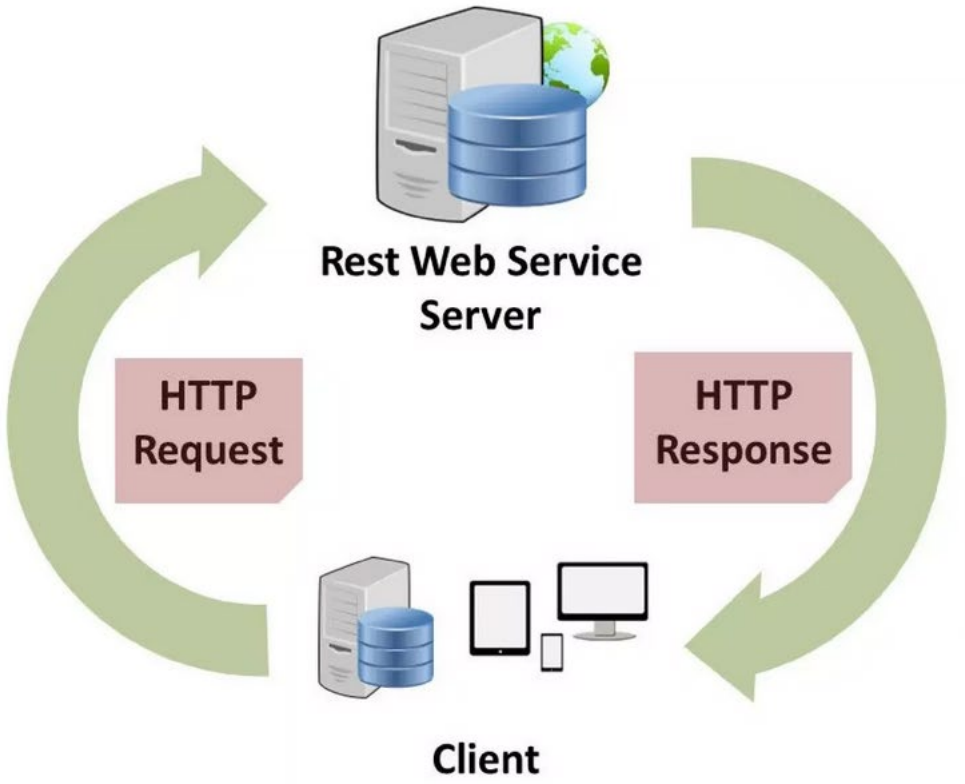
200 - OK	403 - Forbidden
201 - Created	404 - Not found
301 - Moved Permanently	410 - Gone
302 - Found	500 - Internal Server Error
304 - Not Modified	501 - Not Implemented
400 - Bad Request	502 - Bad Gateway
401 - Unauthorized	503 - Service Unavailable

Webservices

- RPC, SOAP, REST

Essencialmente, Rest é **Representational State Transfer** que, em português, é “Transferência de Estado Representacional”.
Por sua definição, trata-se de um conjunto de princípios e definições necessário para a criação de um projeto com interfaces bem definidas.
É, na verdade, uma abstração da arquitetura da Web. Por ser muito requisitada nas equipes de desenvolvimento, é importante se aprofundar nos estudos desse recurso.
O termo foi usado, pela primeira vez, em uma dissertação escrita por Roy Fielding (cientista de computação norte-americano) no ano 2000.

Webservices REST



Client

- Pcs
- Macs
- Mobile devices
- Outras aplicações
- etc..

Webservices REST - Características

- Conjunto de princípios/idéias para criação de Webservices
- Não esta amarrado exclusivamente ao HTTP
- Porém HTTP é amplamente usado e fornece enormes vantagens ao REST
- Uma **API REST** bem desenhada basicamente faz uso:
  - URIs Individuais por recurso.
  - **Verbos HTTP** (POST, PUT, GET e DELETE).
  - Uso apropriado de **Headers HTTP** e **Status code** de resposta de requisições.
  - Múltiplas opções de **formatos de representação** de recursos (**JSON, XML**, etc.)
  - Controles de **hipermídia**.

Webservices REST - Vantagens

- Simplicidade
- Múltiplos tipos de formato de representação (JSON, XML, etc.)
- Parse de JSON é veloz
- JSON tem suporte nativo em Javascript
- JSON pode ser lido e escrito em qualquer linguagem de programação
- Cache
- Facilidade em escalabilidade

Webservices REST - Verbos

Requisição

```
POST /livros
Content-type: application/json
{
  "titulo":"My New Book",
  "autor":"Autor Foo",
  "isbn":"111-11-1111-111-1"
}
```

Resposta

```
HTTP/1.1 201 Created
Host: localhost
Connection: close
Cache-Control: no-cache
Location: /livros/4
```

Requisição

```
PUT /livros/4
Content-type: application/json
{
  "titulo":"My Book",
  "autor":"Autor Qualquer",
  "isbn":"111-11-1111-111-1"
}
```

Resposta

```
HTTP/1.1 200 OK
Host: localhost
Connection: close
Cache-Control: no-cache
Content-type: application/json

{"titulo":"My Book", "autor":"Autor Qualquer",
  "isbn":"111-11-1111-111-1"}
```

Requisição

```
DELETE /livros/4
```

Resposta

```
HTTP/1.1 204 No Content
Host: localhost
Connection: close
```

Requisição

```
PATCH /livros/4
Content-type: application/json
{
  "autor":"Autor XPTO",
}
```

Resposta

```
HTTP/1.1 200 OK
Host: localhost
Connection: close
Cache-Control: no-cache
Content-type: application/json

{"titulo":"My Book", "autor":"Autor XPTO",
  "isbn":"111-11-1111-111-1"}
```

Requisição

```
GET /livros/4
GET /livros
```

Resposta

```
HTTP/1.1 200 OK
Host: localhost
Connection: close
Content-type: application/json

{"titulo":"My New Book", "autor":"Autor Foo",
  "isbn":"111-11-1111-111-1"}
```

Webservices REST - Formatos

Formatos de Representação - Mime Type

JSON

application/json

XML

application/xml

Formatos de Representação - Resposta

JSON

Content-type: application/json

XML

Content-type: application/xml