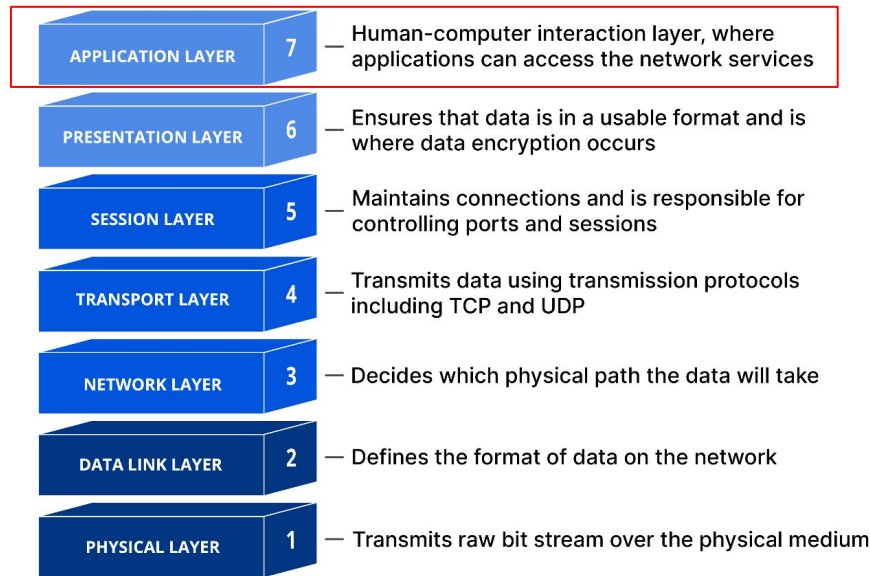


# HTTP

Flutter

# Pilha de protocolos OSI

- Protocolo da camada de aplicação
  - Transmissão de documentos hipermídia (HTML)
    - Imagens
    - Vídeos
    - JSON



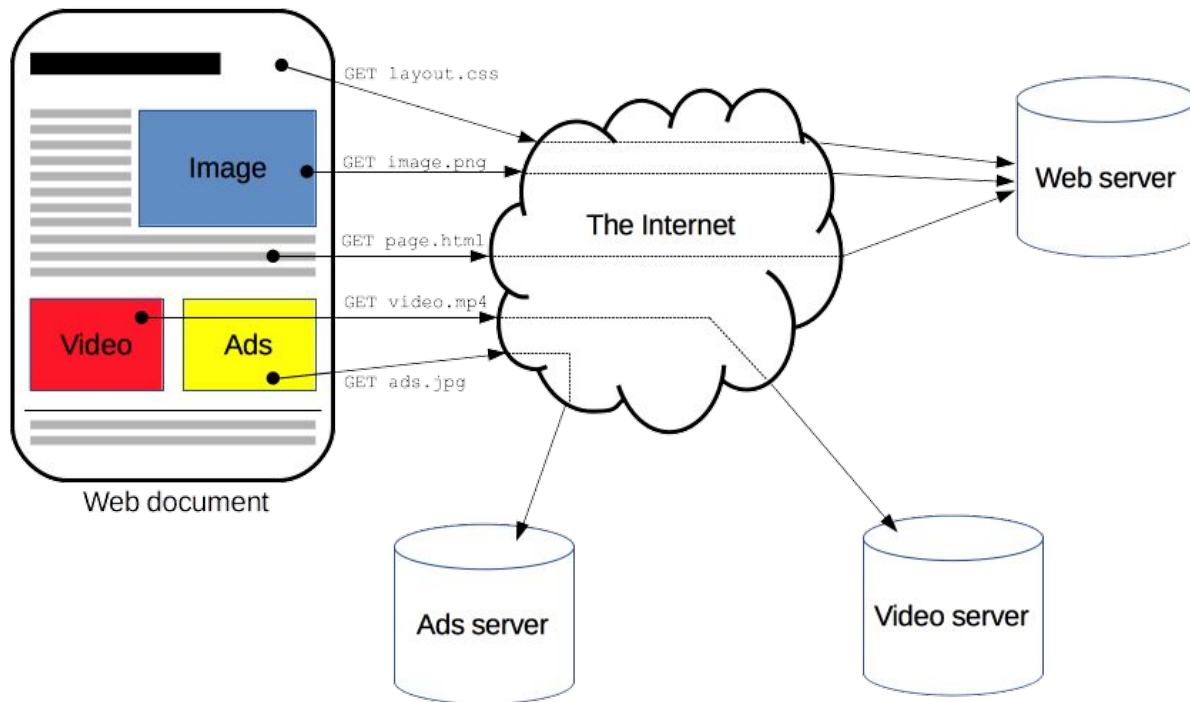
# Visão geral HTTP

- O que é HTTP?
  - Hyper Text Transfer Protocol
- Arquitetura cliente e servidor
  - Requisição e resposta http
- Cliente
  - Browser
    - Safari, Chrome, Firefox, Opera, Edge, ...
    - Qualquer outro software que faz requisição HTTP
- Servidor
  - Computadores na nuvem
    - Nginx, Apache, IIS

# Visão geral HTTP

- Protocolo para obtenção de recursos
  - Cliente abre a conexão (Servidor apenas espera a chamada inicial)
    - Diferente de socket/web socket

# Requisição e resposta



# Cliente HTTP

- User Agent
  - Qualquer ferramenta que age representando o usuário
  - Sempre quem inicia a requisição é o user agent
    - Servidor apenas responde
- Browser, app, software benchmark, sensor, interruptor, geladeira, entre outros

# Servidor HTTP

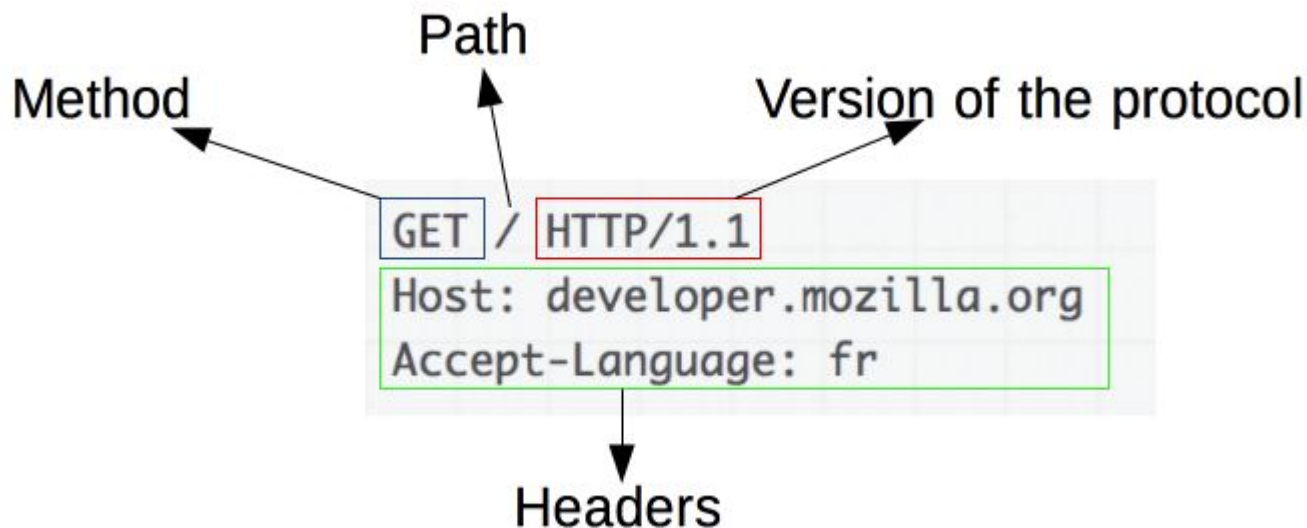
- Serve o que foi solicitado
  - Página HTML, dados de uma API, entre outros
  - Sempre quem inicia a requisição é o user agent
    - Servidor apenas responde

# Requisição e resposta

- Clientes e servidores se comunicam trocando mensagens individuais
  - Geralmente TCP ou TCP criptografado com TLS (HTTPS)
    - UDP pode ser usado: live stream



# Requisição



# Requisição

- Método
  - Define qual operação o cliente quer fazer
  - GET
  - POST
  - PUT
  - PATCH
  - DELETE
  - HEAD
  - OPTIONS
- Path
  - Caminho do recurso a ser buscado
  - Sem o protocolo (http ou https), sem o domínio (google.maps.com), sem a porta (80 ou 443)

# Requisição - Método

- Ação a ser executada em um dado recurso
- Cada um possui a sua semântica
- Método
  - GET
    - Busca a representação de um recurso específico (Busca dados do backend)
  - HEAD
    - Idêntico ao GET, porém vem sem o corpo
  - POST
    - Submete dados à um recurso específico (Criar dados no backend)
  - PUT
    - Substitui os dados do recurso com os novos que vieram na requisição (Substituir dados no backend)
- Método
  - PATCH
    - Similar ao PUT, porém atualizações em campos parciais
  - DELETE
    - Deleta a representação de um recurso específico (Deleta dados do backend)
  - CONNECT
    - Solicita a criação de um tunnel
  - OPTIONS
    - Solicita as opções disponíveis para acessar aquele recurso
  - TRACE
    - Depuração

# Requisição

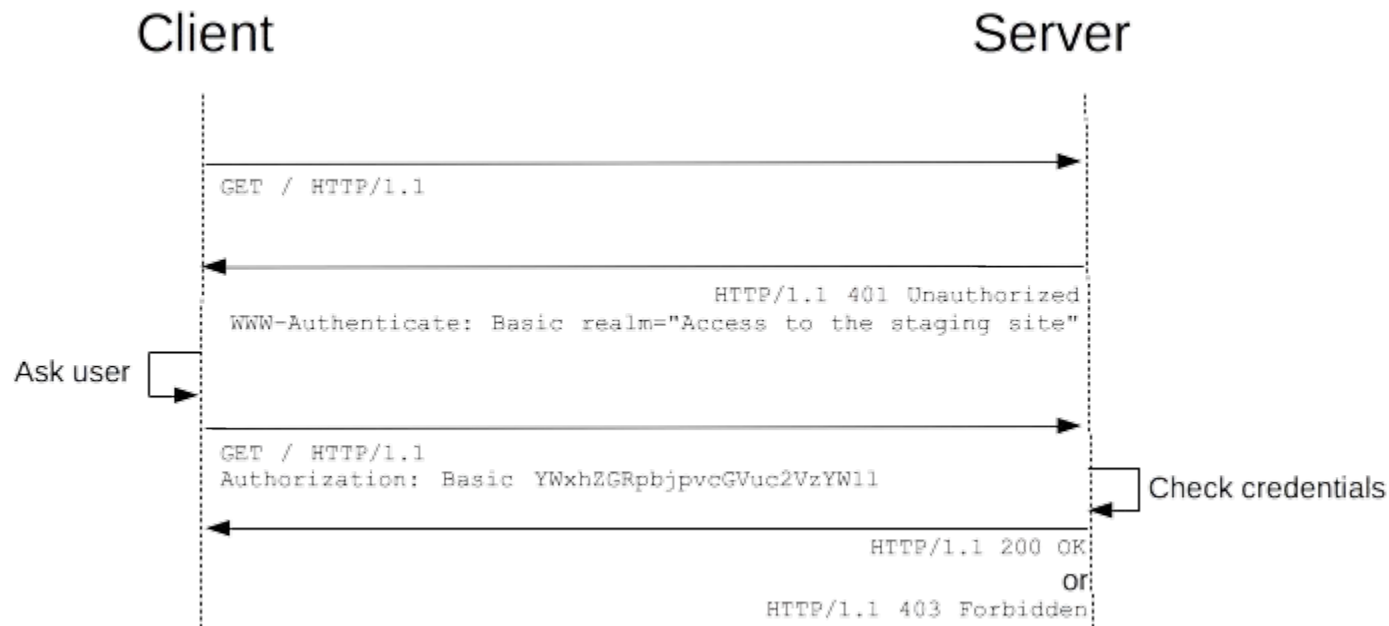
- Versão do protocolo
  - 0.9
  - 1.0
  - 1.1
  - 2.0
- Cabeçalho (opcional)
  - Passa informações adicionais ao servidor
- Corpo
  - Alguns métodos exigem
    - POST

# Requisição - Cabeçalho

- Cabeçalhos

- WWW-Authenticate
  - Define o método de autenticação para acessar o recurso (Bearer, Basic, ...)
- Authorization
  - Define as credenciais para autenticar um user agent
- Content-Type
  - Define o tipo do recurso a ser transmitido
- if-Modified-Since
  - Requisição condicional. Entidade transmitida apenas se foi modificada após data especificada. (cache)

# Requisição - Cabeçalho

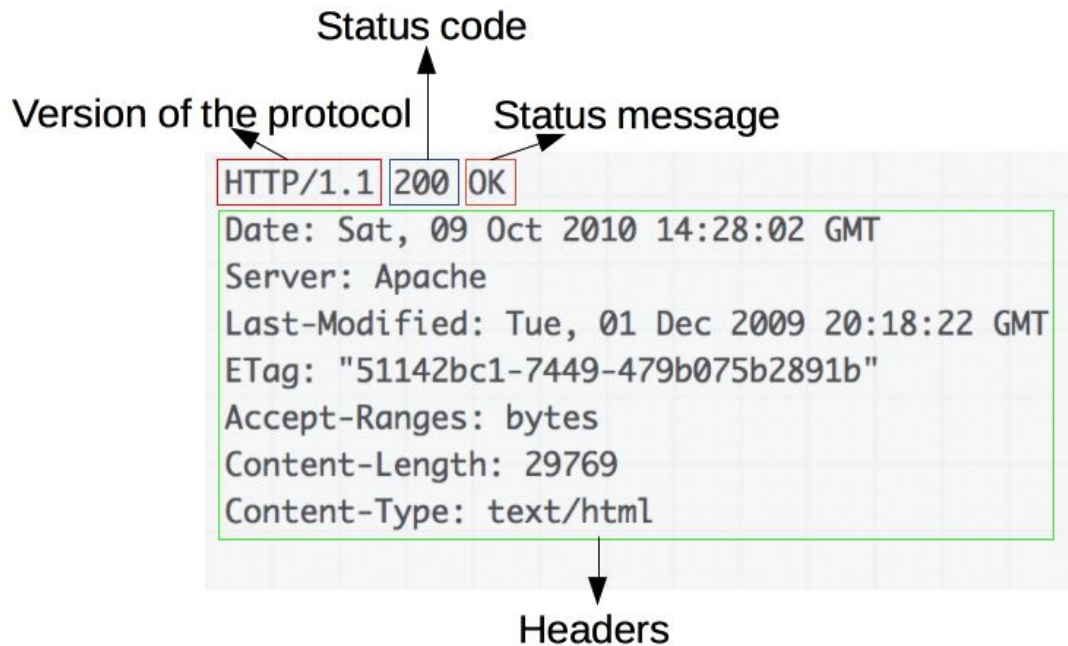


# Requisição



```
GET / HTTP/1.1  
Host: developer.mozilla.org  
Accept-Language: pt-br
```

# Resposta





# Resposta

- Versão do protocolo
- Código de status (status code)
  - Indica se houve sucesso ou não na requisição
  - 1xx
  - 2xx
  - 3xx
  - 4xx
  - 5xx
- Mensagem de estado (status message)
  - Descrição textual do status code
- Cabeçalho
- Corpo
  - Alguns métodos retornam
    - GET

# Resposta - Status code

- Código de status (status code)
  - 100 a 199 - Informações
  - 200 a 299 - Sucesso
  - 300 a 399 - Redirecionamento
  - 400 a 499 - Erro de cliente
  - 500 a 599 - Erro de servidor

# Resposta



```
HTTP/1.1 200 OK
Date: Sat, 09 Oct 2010 14:28:02 GMT
Server: Apache
Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
ETag: "51142bc1-7449-479b075b2891b"
Accept-Ranges: bytes
Content-Length: 29769
Content-Type: text/html

<!DOCTYPE html... (here comes the 29769 bytes of the requested web page)
```

# Características mensagens HTTP

- Humanamente legível
- Fácil para testes
- Extensível
  - Cabeçalhos novos podem ser acordados entre as partes
- Stateless
  - Carrinho de compras
    - Salvo no backend
    - Cookies no client

# Fluxo requisição e resposta

1. Abertura de uma conexão TCP
2. Cliente envia mensagem HTTP
3. Cliente recebe mensagem do servidor
4. Fecha a conexão

# Padrão para transferência dos dados

- JSON
- XML

XML

vs.

JSON

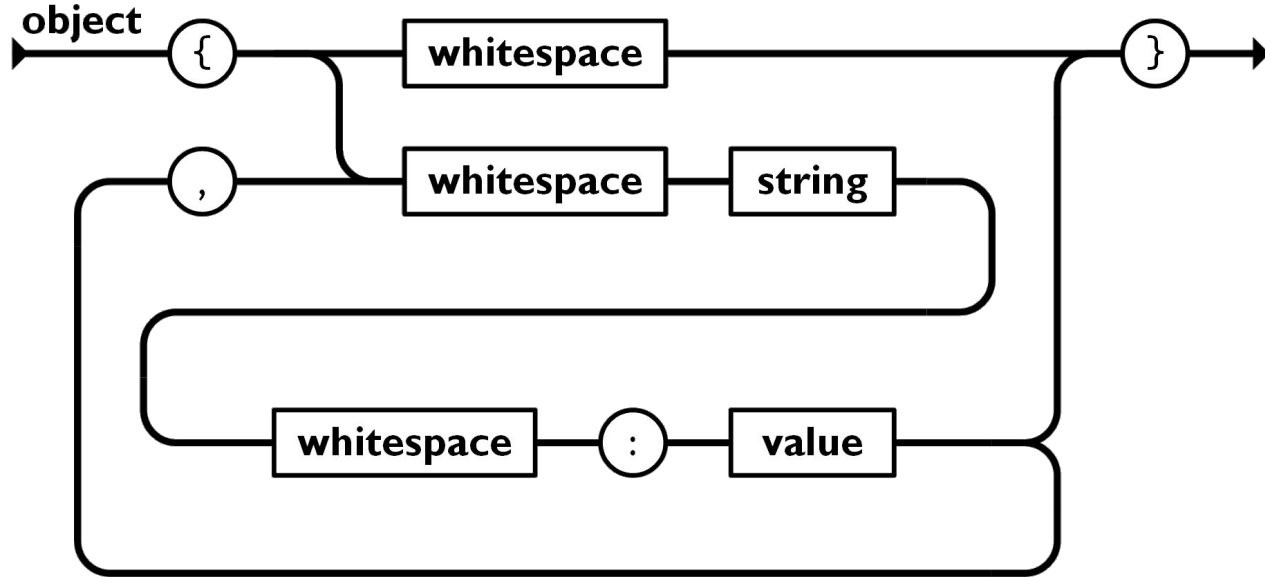
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <endereco>
3   <cep>31270901</cep>
4   <city>Belo Horizonte</city>
5   <neighborhood>Pampulha</neighborhood>
6   <service>correios</service>
7   <state>MG</state>
8   <street>Av. Presidente Antônio Carlos, 6627</street>
9 </endereco>
```

```
1 {
2   "endereco": {
3     "cep": "31270901",
4     "city": "Belo Horizonte",
5     "neighborhood": "Pampulha",
6     "service": "correios",
7     "state": "MG",
8     "street": "Av. Presidente Antônio Carlos, 6627"
9   }
10 }
```

# JSON

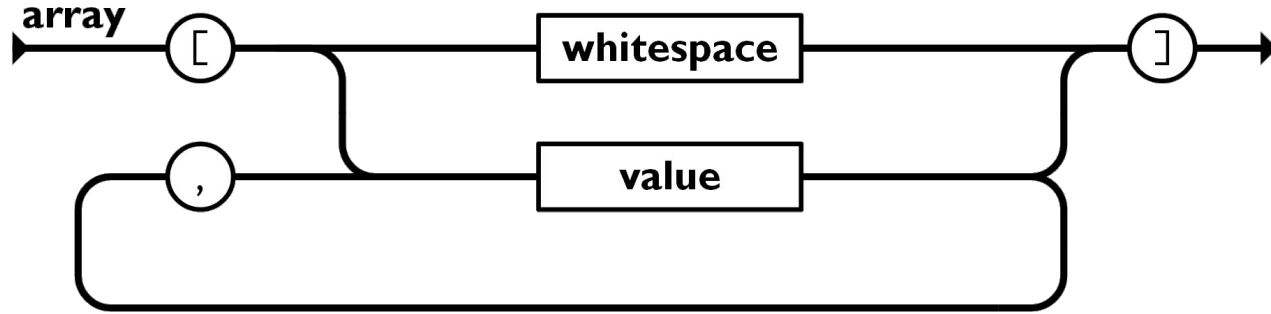
- JavaScript Object Notation
- Padrão leve para transferência de dados
- Humanamente legível
- Fácil interpretação e geração
- Formato de texto
- Independente de linguagem
  - Todas suportam nativamente ou por meio de libs externas

# JSON - Objeto

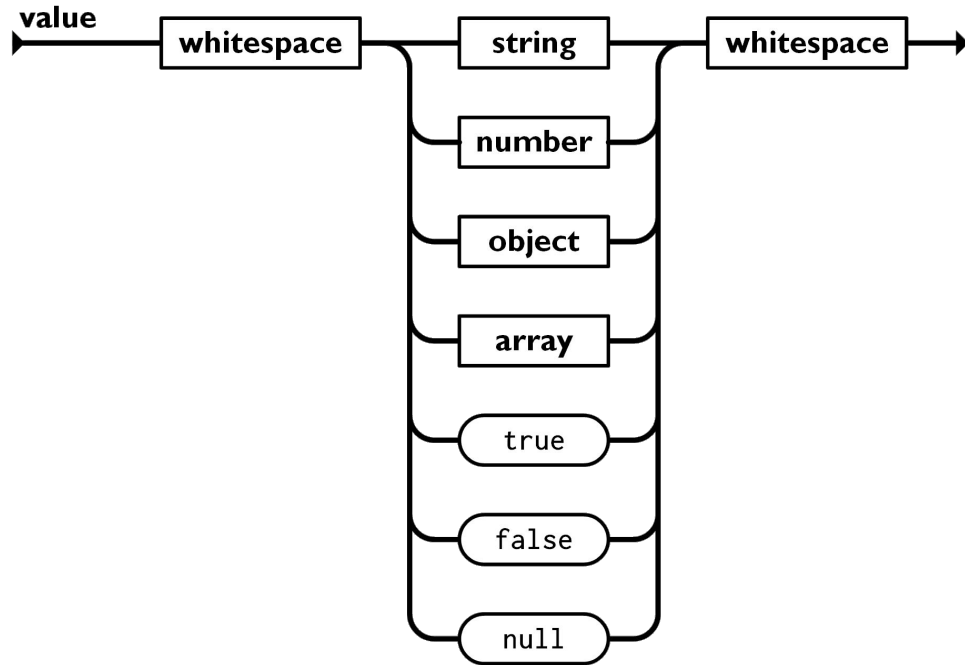




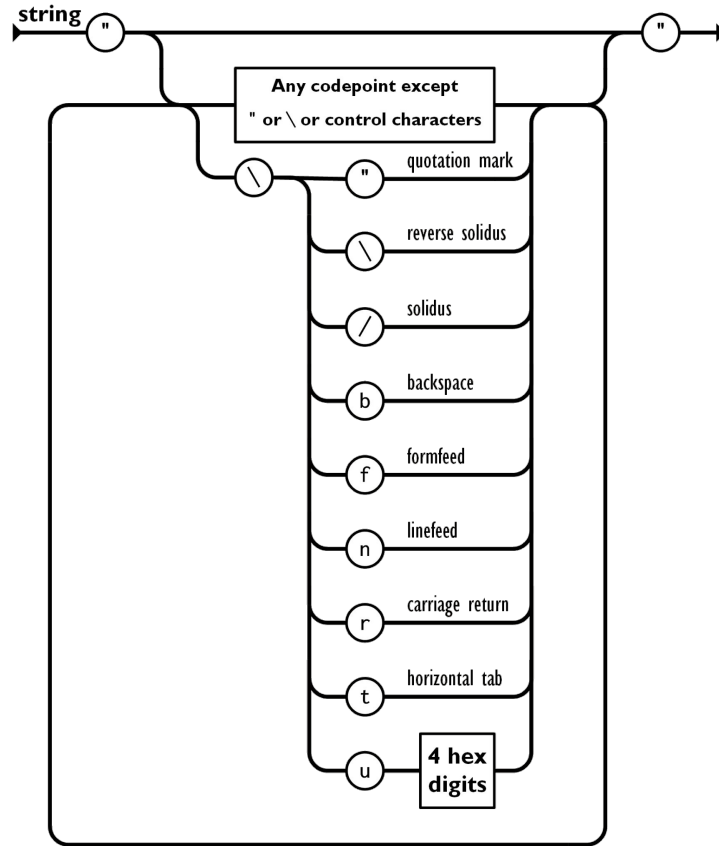
# JSON - Array



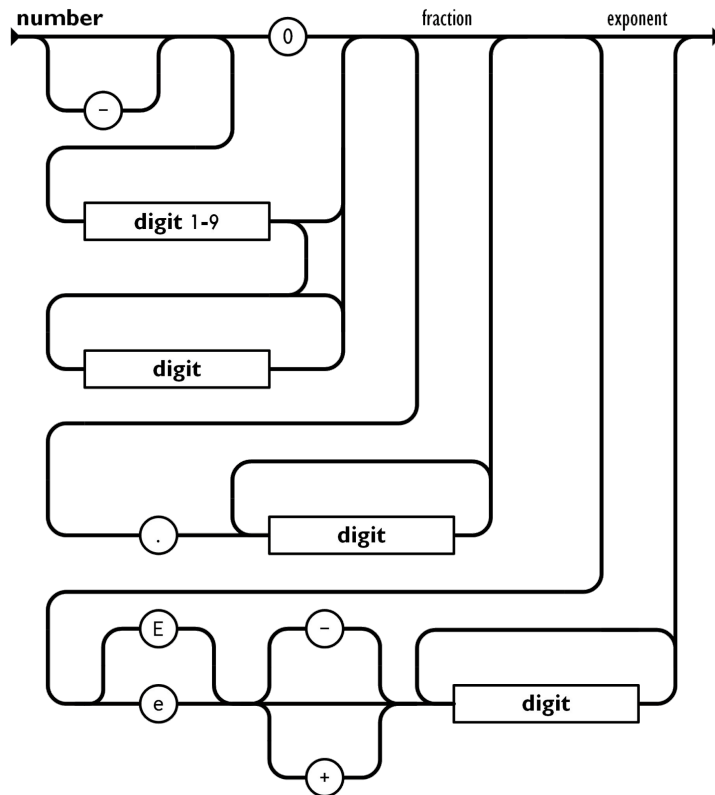
# JSON - Value



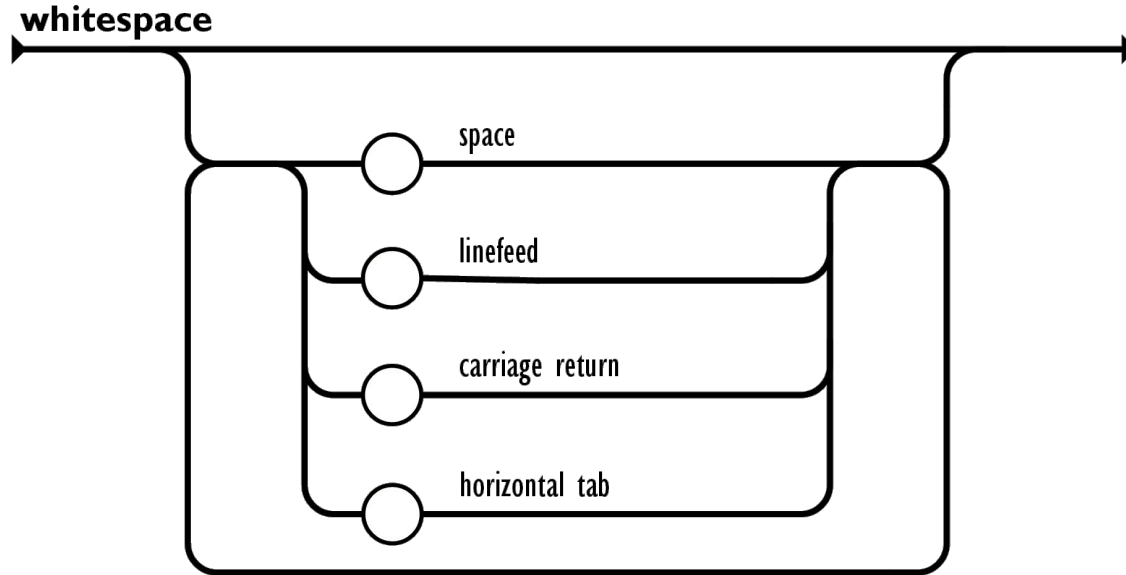
# JSON - String



# JSON - Número



# JSON - Espaçamento





# http 1.1.0

Published 57 days ago •  dart.dev Dart 3 compatible

[SDK](#) [DART](#) [FLUTTER](#) [PLATFORM](#) [ANDROID](#) [IOS](#) [LINUX](#) [MACOS](#) [WEB](#) [WINDOWS](#)

 6.6K

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Versions](#) [Scores](#)

 **v1.1.0**  dart.dev

A composable, Future-based library for making HTTP requests.

This package contains a set of high-level functions and classes that make it easy to consume HTTP resources. It's multi-platform, and supports mobile, desktop, and the browser.

## Using

The easiest way to use this library is via the top-level functions. They allow you to make individual HTTP requests with minimal hassle:

**6646** **140** **100%**  
LIKES PUB POINTS POPULARITY

Publisher

 [dart.dev](#)

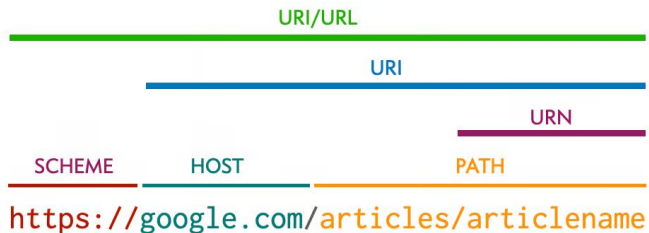
Metadata

A composable, multi-platform, Future-based API for HTTP requests.

[Repository \(GitHub\)](#)

[View/report issues](#)

# http URI



DANIEL MIESSLER 2022

```
var httpsUri = Uri(  
  scheme: 'https',  
  host: 'example.com',  
  path: '/page',  
  queryParameters: {'search': 'blue', 'limit': '10'});  
print(httpsUri); // https://example.com/page/?search=blue&limit=10  
//ou  
httpsUri = Uri.https('example.com', 'page', {'search': 'blue', 'limit': '10'});  
print(httpsUri); // https://example.com/api/fetch?limit=10  
//ou  
httpsUri = Uri.parse('https://example.com/page?search=blue&limit=10');  
print(httpsUri); // https://example.com/api/fetch?limit=10
```

# dart:convert lib

## jsonEncode function

```
String jsonEncode(  
  Object? object,  
  {Object? toEncodable(  
    Object? nonEncodable  
  )?}  
)
```

Converts object to a JSON string.

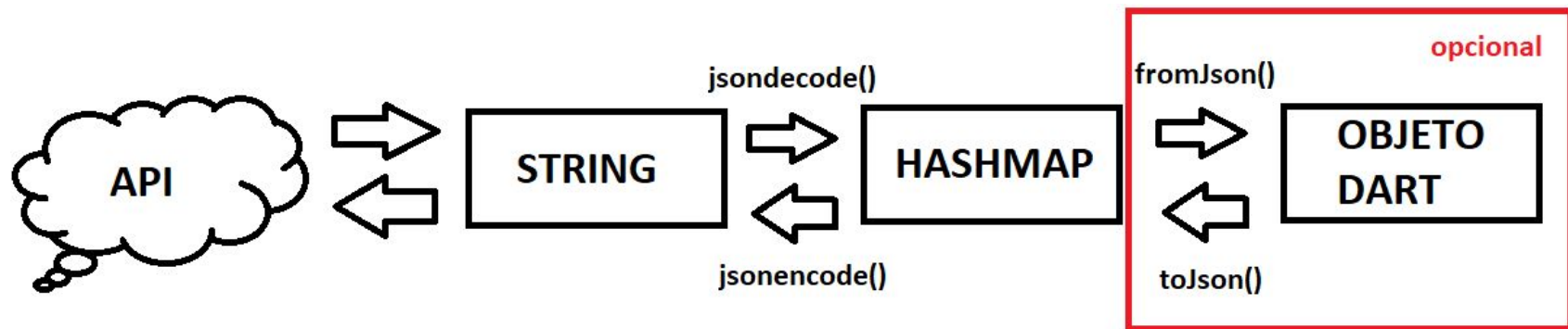
## jsonDecode function

```
dynamic jsonDecode(  
  String source,  
  {Object? reviver(  
    Object? key,  
    Object? value  
  )?}  
)
```

Parses the string and returns the resulting Json object.



# dart:convert lib



# dart:convert lib



```
var encoded = json.encode({ "name": "Java","age":48 });
var decoded = json.decode('{ "name": "Java","age":48 }');
print('${encoded.runtimeType} - ${encoded}');//String - {"name":"Java","age":48}
print('${decoded.runtimeType} - ${decoded}');//_JsonMap - {name: Java, age: 48}
print('${decoded['age'].runtimeType} - ${decoded['age']}');//int - 48
```



```
var encoded = json.encode([1, 2, { "name": "Java","age":48 }]);
var decoded = json.decode('[1, 2, { "name": "Java","age":48 }]');
print('${encoded.runtimeType} - ${encoded}');//String - [1,2,{"name":"Java","age":48}]
print('${decoded.runtimeType} - ${decoded}');//JSArray<dynamic> - [1, 2, {name: Java, age: 48}]
print('${decoded[2].runtimeType} - ${decoded[2]}');//_JsonMap - {name: Java, age: 48}
```

# http lib

`delete(Uri url, {Map<String, String>? headers, Object? body, Encoding? encoding}) → Future<Response>`

Sends an HTTP DELETE request with the given headers to the given URL.

`get(Uri url, {Map<String, String>? headers}) → Future<Response>`

Sends an HTTP GET request with the given headers to the given URL.

`head(Uri url, {Map<String, String>? headers}) → Future<Response>`

Sends an HTTP HEAD request with the given headers to the given URL.

`patch(Uri url, {Map<String, String>? headers, Object? body, Encoding? encoding}) → Future<Response>`

Sends an HTTP PATCH request with the given headers and body to the given URL.


`post(Uri url, {Map<String, String>? headers, Object? body, Encoding? encoding}) → Future<Response>`

Sends an HTTP POST request with the given headers and body to the given URL.

`put(Uri url, {Map<String, String>? headers, Object? body, Encoding? encoding}) → Future<Response>`

Sends an HTTP PUT request with the given headers and body to the given URL.

# http lib



```
import 'package:http/http.dart' as http;

var url = Uri.https('myapi', 'users');
var response = await http.post(url, body: {'name': 'Joseph', 'color': 'blue'});
print('Response status: ${response.statusCode}');
print('Response body: ${response.body}');
```



```
Response response = await post(url,
  headers: {
    'authorization': 'Bearer GDSU3J4JV_fsd',
    'Content-Type': 'application/json'
  },
  body: '{"name": "Dart"}'
);
```

# http lib

## Response class

An HTTP response where the entire response body is known in advance.

Inheritance

[Object](#) > [BaseResponse](#) > Response

## Constructors

`Response(String body, int statusCode, {BaseRequest? request, Map<String, String> headers = const {}, bool isRedirect = false, bool persistentConnection = true, String? reasonPhrase})`

Creates a new HTTP response with a string body.

`Response.bytes(List<int> bodyBytes, int statusCode, {BaseRequest? request, Map<String, String> headers = const {}, bool isRedirect = false, bool persistentConnection = true, String? reasonPhrase})`

Create a new HTTP response with a byte array body.

# http lib

## Properties

*body* → String

The body of the response as a string.

read-only

*bodyBytes* → Uint8List

The bytes comprising the body of this response.

final

*contentLength* → int?

The size of the response body, in bytes.

final inherited

*hashCode* → int

The hash code for this object.

read-only inherited

*headers* → Map<String, String>

final inherited

*isRedirect* → bool

final inherited

*isRedirect* → bool

final inherited

*persistentConnection* → bool

Whether the server requested that a persistent connection be maintained.

final inherited

*reasonPhrase* → String?

The reason phrase associated with the status code.

final inherited

*request* → BaseRequest?

The (frozen) request that triggered this response.

final inherited

*runtimeType* → Type

A representation of the runtime type of the object.

read-only inherited

*statusCode* → int

The HTTP status code for this response.

final inherited



# http lib

```
import 'dart:convert' as convert;

import 'package:http/http.dart' as http;

void main(List<String> arguments) async {
  // This example uses the Google Books API to search for books about http.
  // https://developers.google.com/books/docs/overview
  var url =
    Uri.https('www.googleapis.com', '/books/v1/volumes', {'q': '{http}'});

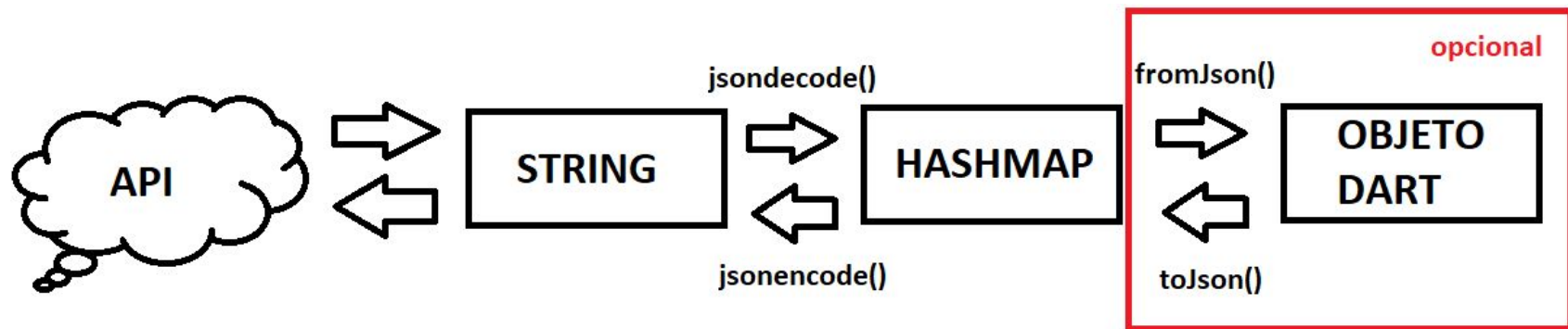
  // Await the http get response, then decode the json-formatted response.
  var response = await http.get(url);
  if (response.statusCode == 200) {
    var jsonResponse =
      convert.jsonDecode(response.body) as Map<String, dynamic>;
    var itemCount = jsonResponse['totalItems'];
    print('Number of books about http: $itemCount.');
  } else {
    print('Request failed with status: ${response.statusCode}.');
  }
}
```

# http lib

```
{
  "kind": "books#volumes",
  "items": [
    {
      "kind": "books#volume",
      "id": "zaRoX10_UsMC",
      "etag": "pm1sLMgKfMA",
      "selfLink": "https://www.googleapis.com/books/v1/volumes/zaRoX10_UsMC",
      "volumeInfo": {
        "title": "Flowers",
        "authors": [
          "Paul McEvoy"
        ],
        ...
      },
      //.... mais objetos
    }
  ],
  "totalItems": 12
}
```



# dart:convert lib



# http lib

## JSON to Dart

Paste your JSON in the textarea below, click convert and get your Dart classes for free.

JSON

```
1 {  
2   "items": [  
3     {  
4       "title": "Algoritmos: Teoria e Pr  
5       "authors": [  
6         "Thomas H. Cormen",  
7         "Charles Eric L. iserson",  
8         "Ronald Rivest",  
9         "Clifford Stein"  
10    ]  
11  },  
12 ],  
13 "totalItems": 3  
14 }
```

GoogleBooks

Generate Dart

☐ Use private fields

Copy Dart code to clipboard

```
class GoogleBooks {  
  List<Items>? items;  
  int? totalItems;  
  
  GoogleBooks({this.items, this.totalItems});  
  
  GoogleBooks.fromJson(Map<String, dynamic> json) {  
    if (json['items'] != null) {  
      items = <Items>[];  
      json['items'].forEach((v) {  
        items!.add(new Items.fromJson(v));  
      });  
    }  
    totalItems = json['totalItems'];  
  }  
  
  Map<String, dynamic> toJson() {  
    final Map<String, dynamic> data = new Map<String, dynamic>();  
    if (this.items != null) {  
      data['items'] = this.items!.map((v) => v.toJson()).toList();  
    }  
    if (this.totalItems != null) {  
      data['totalItems'] = this.totalItems;  
    }  
    return data;  
  }  
}
```

# http lib

```
{
  "name": "Dart",
  "phone": [
    {
      "ddd": "84",
      "number": "9.9876-0000"
    }
  ]
}
```

Gerar Classes com json to dart

# Referências

- <https://www.json.org/json-pt.html>
- [https://www.ecma-international.org/wp-content/uploads/ECMA-404\\_2nd\\_edition\\_december\\_2017.pdf](https://www.ecma-international.org/wp-content/uploads/ECMA-404_2nd_edition_december_2017.pdf)
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>
- <https://danielmiessler.com/p/difference-between-uri-url/>
-