



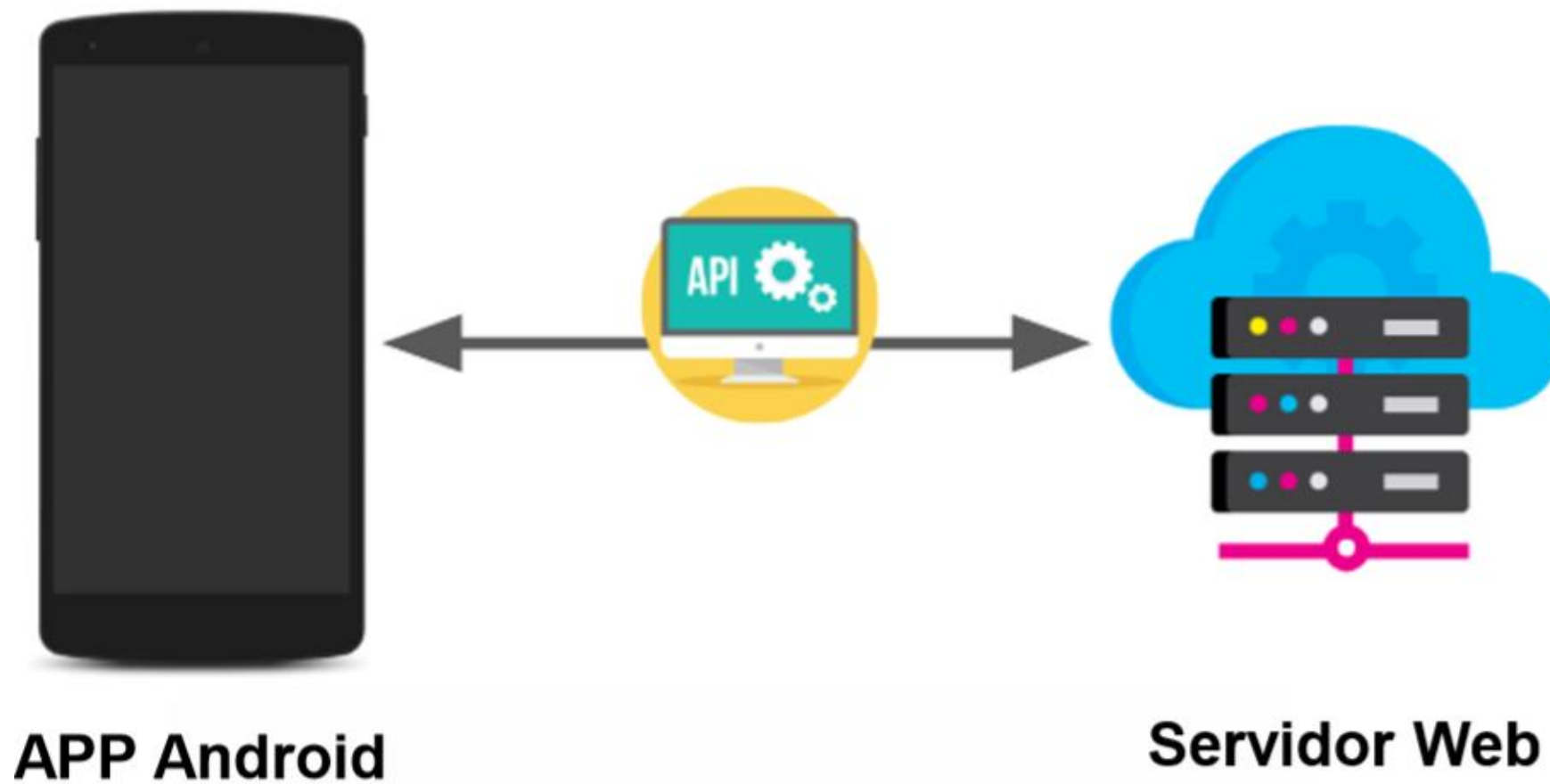
Web Services

Prof. Ilo Rivero
(ilo@pucminas.br)

O que vamos aprender nessa aula?

- Nessa aula vamos aprender a consumir um **serviço na web**, através de requisições.

Introdução



Web API

- Uma Web API é uma interface de programação de aplicações (Application Programming Interface - API)
- Podemos nos conectar a uma API por meio de uma aplicação ou pelo próprio browser

- Motivos para usar uma API:
 - O aplicativo se torna mais simples (sem consultas SQL, focada nos dados)
 - O banco de dados fica isolado
 - Toda lógica de negócio fica dentro da API
 - Segurança

Requisições HTTP

- HTTP é um acrônimo para *HyperText Transfer Protocol*, ou Protocolo de Transferência de HiperTexto
- Padronizou a comunicação entre navegadores e servidores
- Tipos de requisição:
 - **Get:** recuperar dados no servidor
 - **Post:** criar dados no servidor
 - **Put:** atualizar dados no servidor
 - **Delete:** deletar dados no servidor

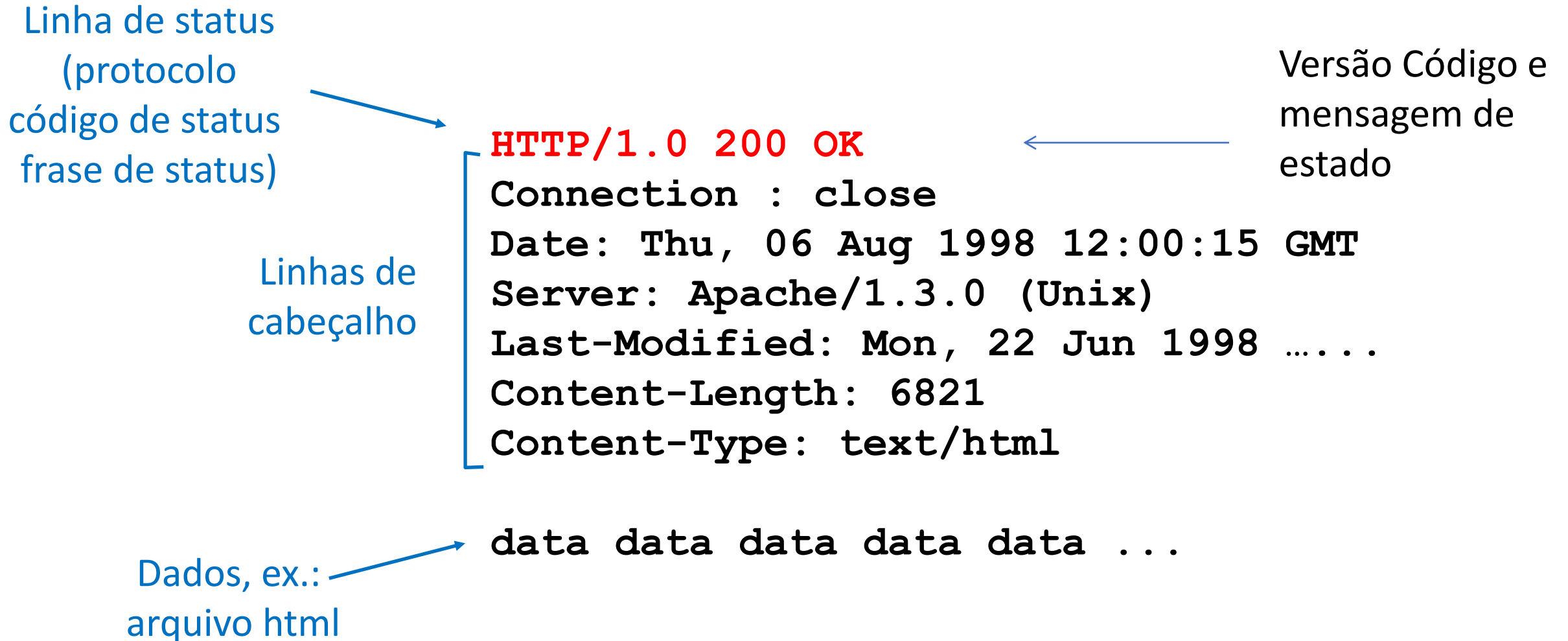
HTTP

- Utiliza TCP
- Cliente inicia conexão TCP (cria socket) para o servidor na porta 80
- Servidor aceita uma conexão TCP do cliente
- mensagens HTTP (mensagens do protocolo de camada de aplicação) são trocadas entre o browser (cliente HTTP) e o servidor Web (servidor HTTP)

HTTP

- O TCP provê ao HTTP um serviço confiável de transferência de dados
- **Vantagem:** o HTTP não precisa se preocupar com dados perdidos
- A conexão TCP é fechada

HTTP Response



HTTP Response

Linha de status
(protocolo
código de status
frase de status)

Linhas de
cabeçalho

Data da ultima modificação do
objeto no servidor (útil para
informações de cache)

HTTP/1.0 200 OK

Connection : close

Date: Thu, 06 Aug 1998 12:00:15 GMT

Server: Apache/1.3.0 (Unix)

Last-Modified: Mon, 22 Jun 1998

Content-Length: 6821

Content-Type: text/html

Dados, ex.:
arquivo html

data data data data data ...

JSON

- JavaScript Object Notation (Notação de Objetos JavaScript): é uma formatação leve de troca de dados
- Para seres humanos, é fácil de ler e escrever
- Para máquinas, é fácil de interpretar e gerar
- É baseado em um subconjunto da linguagem de programação JavaScript

JSON

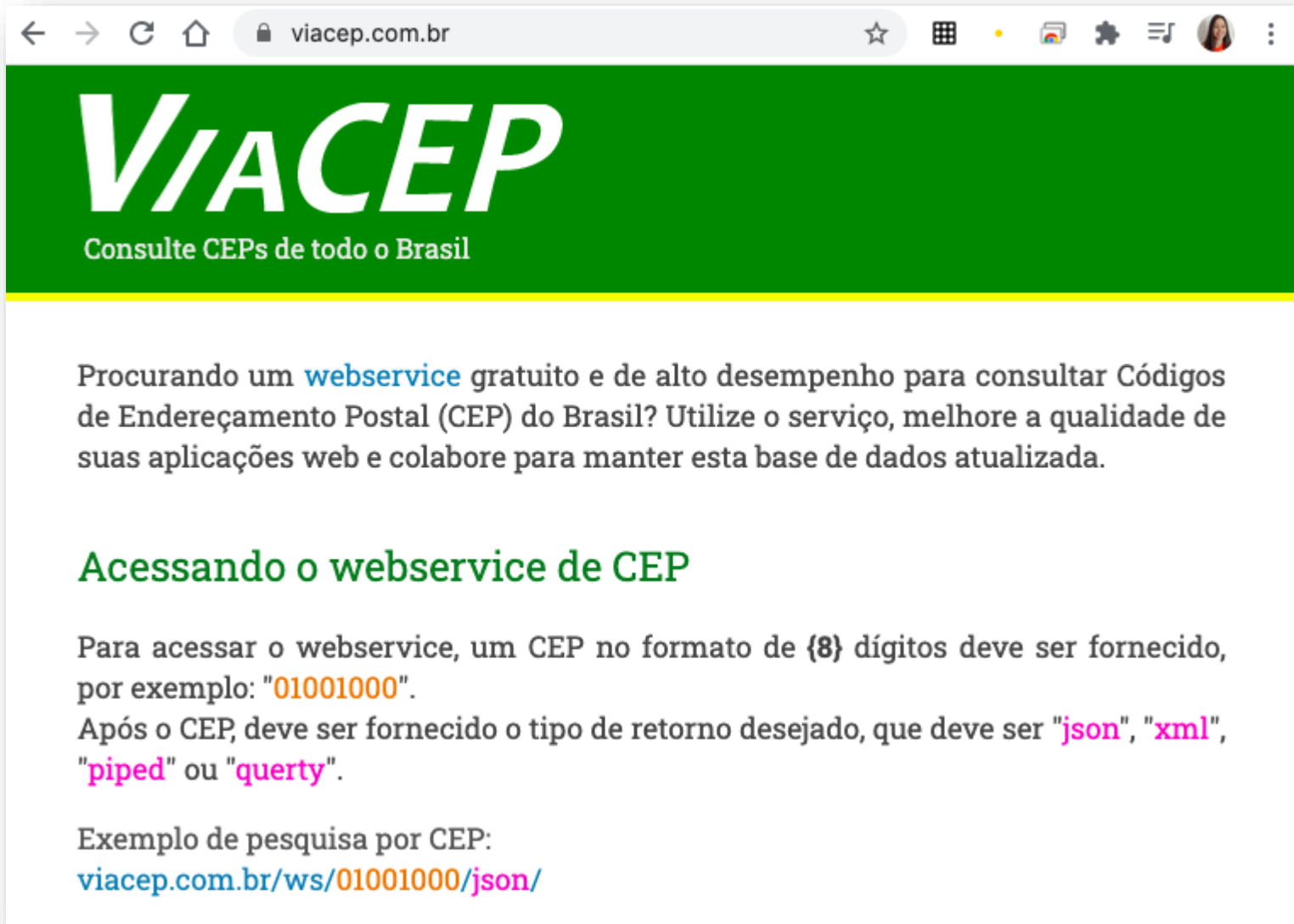
- Em JSON, os dados são apresentados:
- Um objeto é um conjunto desordenado de pares nome/valor
- Um objeto começa com {chave de abertura e termina com }chave de fechamento. Cada nome é seguido por :dois pontos e os pares nome/valor são seguidos por , vírgula.

JSON

```
01.  {
02.    "Androides": [{
03.      "nome": "Nougat",
04.      "versao": 7.0
05.    }, {
06.      "nome": "Marshmallow",
07.      "versao": 6.0
08.    }, {
09.      "nome": "Lollipop",
10.      "versao": 5.0
11.    }]
12.  }
```

Web Service - Via Cep

- Para elaborar nosso exemplo, usaremos o webservice viacep.com.br, que é gratuito e de alto desempenho



Dependência HTTP

http 0.13.3

Published May 3, 2021 •  dart.dev Null safety

[DART](#) [NATIVE](#) [JS](#)
[FLUTTER](#) [ANDROID](#) [IOS](#) [LINUX](#) [MACOS](#) [WEB](#) [WINDOWS](#)

[Readme](#) [Changelog](#) [Example](#) [Installing](#) [Version](#)

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

pub v0.13.3  Dart CI passing

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.

This will add a line like this to your package's pubspec.yaml (and run an implicit `dart pub get`):

```
dependencies:  
  http: ^0.13.3
```

Alternatively, your editor might support `dart pub get` or `flutter pub get`. Check the docs for your editor to learn more.

Import it

Now in your Dart code, you can use:

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

Dependência HTTP

```
dependencies:  
  http: ^0.13.3  
  flutter:  
    sdk: flutter
```

pubspec.yaml,
verificar a versão no
pub.dev

```
home.dart x  
1 import 'package:flutter/material.dart';  
2 import 'package:http/http.dart';
```

Além de incluir essa
dependência, não esqueça de
incluir a biblioteca, usando o
import
'package:http/http.dart';

HTTP

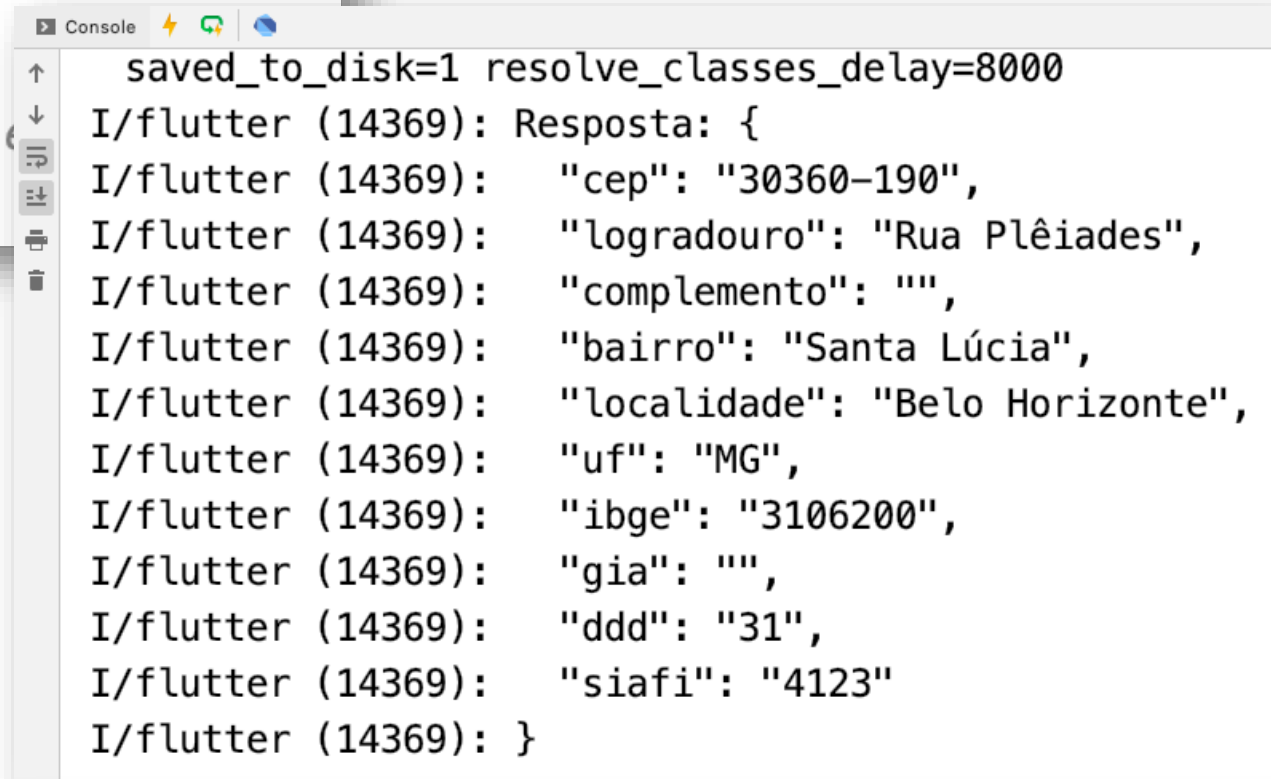
```
import 'package:flutter/material.dart';  
import 'package:http/http.dart' as http;
```

Importar o pacote http, e já renomeando para http, usando “as”, isso facilitará o acesso aos métodos

```
http.Response response;  
response = await http.get(url);
```

HTTP - Exemplo 1

```
_recuperaCep() async{  
  String cep = "30360190";  
  String url = "https://viacep.com.br/ws/${cep}/json/";  
  http.Response response;  
  response = await http.get(url);  
  print("Resposta: " + response.body);  
}
```



The screenshot shows a Flutter console window with the following output:

```
saved_to_disk=1 resolve_classes_delay=8000  
I/flutter (14369): Resposta: {  
I/flutter (14369):   "cep": "30360-190",  
I/flutter (14369):   "logradouro": "Rua Plêiades",  
I/flutter (14369):   "complemento": "",  
I/flutter (14369):   "bairro": "Santa Lúcia",  
I/flutter (14369):   "localidade": "Belo Horizonte",  
I/flutter (14369):   "uf": "MG",  
I/flutter (14369):   "ibge": "3106200",  
I/flutter (14369):   "gia": "",  
I/flutter (14369):   "ddd": "31",  
I/flutter (14369):   "siafi": "4123"  
I/flutter (14369): }
```

Json

```
_recuperaCep() async{  
  String cep = "30360190";  
  String url = "https://viacep.com.br/ws/${cep}/json/";  
  http.Response response;  
  response = await http.get(url);  
  Map<String, dynamic> retorno = json.decode(response.body);  
  String logradouro = retorno["logradouro"];  
  String complemento = retorno["complemento"];  
  String bairro = retorno["bairro"];  
  String localidade = retorno["localidade"];  
  print("Logradouro: ${logradouro} "  
    " complemento: ${complemento}"  
    " bairro: ${bairro}"  
    " localidade: ${localidade}");  
}
```

Para conseguir recuperar um objeto do tipo json será necessário importar outro pacote => **import 'dart:convert';**


Console

saved_to_disk=1 resolve_classes_delay=0000

I/flutter (27834): Logradouro: Rua Plêiades complemento: bairro: Santa Lúcia
localidade: Belo Horizonte

HTTP - Exemplo 1

```
var uri = Uri.parse("https://viacep.com.br/ws/${cepDigitado}/json/");  
http.Response response;  
response = await http.get(uri);
```



Agora o `http.get()` recebe como parâmetro uma URI, sendo assim, é necessário converter a URL em um URI, usando `Uri.parse()`

Exemplo 1

```
_recuperaCep() async{
  String cepDigitado = _controllercep.text;
  var uri = Uri.parse("https://viacep.com.br/ws/${cepDigitado}/json/");
  http.Response response;
  response = await http.get(uri);
  Map<String, dynamic> retorno = json.decode(response.body);
  String logradouro = retorno["logradouro"];
  String complemento = retorno["complemento"];
  String bairro = retorno["bairro"];
  String localidade = retorno["localidade"];
  setState(() { //configurar o _resultado
    _resultado = "${logradouro}, ${complemento}, ${bairro}, ${localidade} ";
  });
}
```

11:28

Consumo de serviço web

Digite o cep ex: 30360190

Clique aqui

Resultado

23:17

Consumo de serviço web

Digite o cep ex: 30360190

32341070

Clique aqui

Rua Moata, , Novo Eldorado, Contagem

1	2	3	-
4	5	6	_
7	8	9	✕
,	0	.	✓