

**INSTITUTO  
FEDERAL**  
Santa Catarina

# Fundamentos de Sistemas para Internet



# Tópicos

- 1) O protocolo HTTP
- 2) Servidores HTTP
- 3) Mensagens HTTP



1/3

# O protocolo HTTP



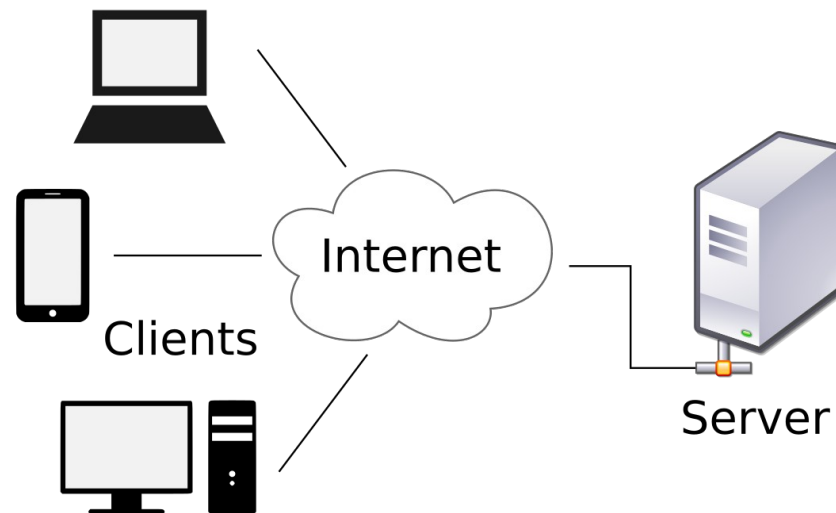
# O protocolo HTTP

- Antes de tudo... o que é um protocolo?
- Em redes/internet, um protocolo pode ser entendido resumidamente como uma série regras para que os dispositivos se entendam;
- É um conjunto de regras e especificações;
- De maneira simples, um protocolo pode ser definido como "as regras que governam" a sintaxe, semântica e sincronização da comunicação. Os protocolos podem ser implementados pelo hardware, software ou por uma combinação dos dois.

# O protocolo HTTP

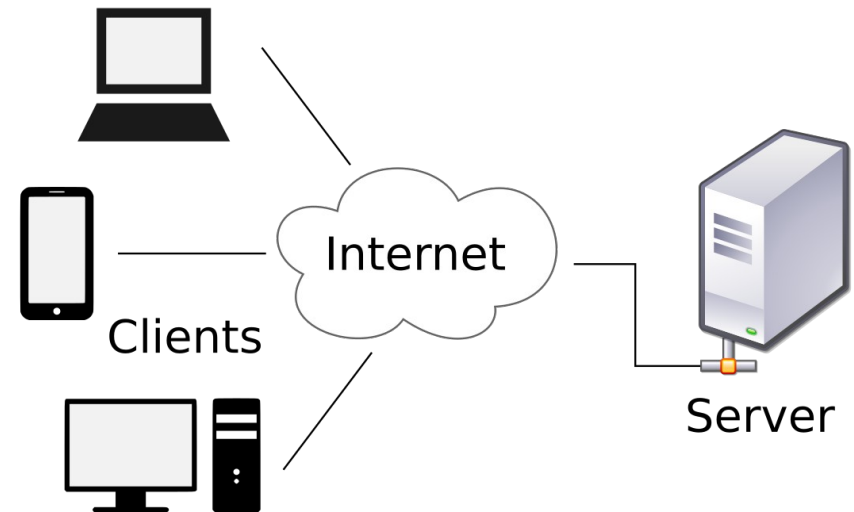
## O que é um servidor HTTP?

- É um servidor que aceita requisições HTTP, geralmente de navegadores, e envia uma resposta de volta, opcionalmente com dados, documentos tais como páginas HTML, imagens, etc.

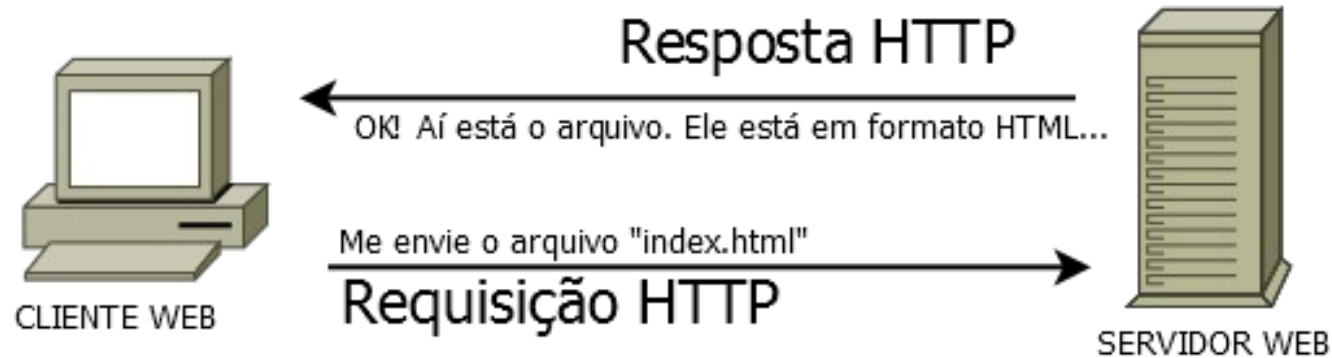


# O protocolo HTTP

- O Hypertext Transfer Protocol (HTTP) é a base para a comunicação da World Wide Web;
- É um protocolo de requisição-resposta entre cliente e servidor;
- É um protocolo de transferência de arquivos do servidor para o cliente;
- É connectionless e stateless.
- Se utiliza de protocolos de transmissão confiável como o TCP



# O protocolo HTTP



- Os clientes HTTP são os navegadores, como Opera, Firefox, Edge e Chrome e fazem as requisições HTTP.
- Os servidores web recebem as requisições, localizam o recurso desejado e enviam de volta através de uma resposta HTTP

# O protocolo HTTP

- A resposta do servidor pode entregar qualquer tipo de recurso, como textos, vídeos, imagens ou documentos pdf.
- Quando este recurso é enviado, o servidor rotula com informações sobre seu tipo. Este rótulo é chamado de MIME (Multipurpose Internet Mail Extension).
- Quando o servidor envia algum recurso, ele anexa um texto contendo o MIME para que o navegador saiba o que fazer com ele (exibir, fazer download, tocar uma música, etc).



# O protocolo HTTP

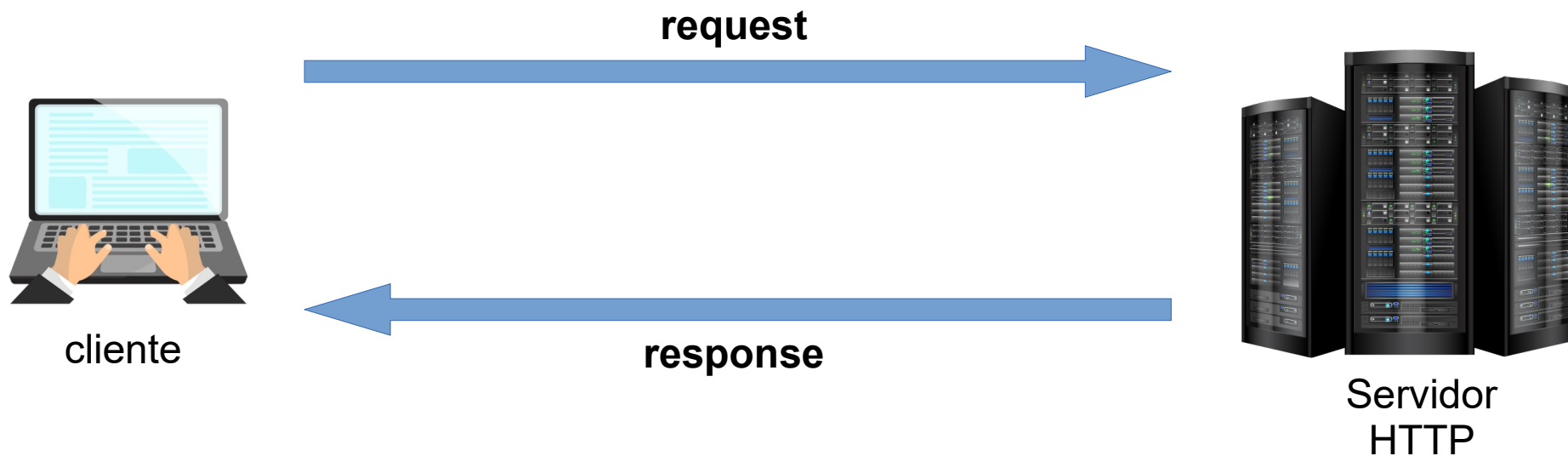
- O MIME possui duas partes separadas por uma barra, indicando o tipo e seu subtipo.
- Exemplos:
  - Arquivos HTML: text/html
  - Arquivos de texto puro: text/plain
  - Imagens JPEG: image/jpeg
  - Imagens GIF: image/gif
  - Vídeo MP4: vídeo/mp4

2/3

# Servidores HTTP



# Servidores HTTP



# Servidores HTTP

- Alguns dos principais servidores são:
  - Apache
  - Nginx
  - IIS
  - Tomcat
  - Node.js



# Servidores HTTP

- Quando você acessa um site pela sua URL (ex: ifsc.edu.br), você está fornecendo o endereço mas não o recurso que gostaria.
- Neste caso, o servidor procurará o especificado em sua configuração de “Directory Index, ou seja, o recurso quer será considerado o índice do site (página inicial)
- Por padrão, a página inicial é a **index.html** (ou outra extensão).

# Servidores HTTP

- Quando você acessa um site pela sua URL (ex: ifsc.edu.br), você está fornecendo o endereço mas não o recurso que gostaria.
- Neste caso, o servidor procurará o especificado em sua configuração de “Directory Index”, ou seja, a pasta e o recurso quer será considerado o índice do site (página inicial)
- Por padrão, a página inicial é a **index.html** (ou outra extensão de acordo com a linguagem server side).

# Servidores HTTP - URLs

- `http://www.ifsc.edu.br/foo/bar.jpg`

protocolo


diretório

recurso

DNS (Domain Name System) do  
host que desejamos acessar o  
recurso;

# Servidores HTTP - URLs

- `https://www.google.com/search?q=ifsc`

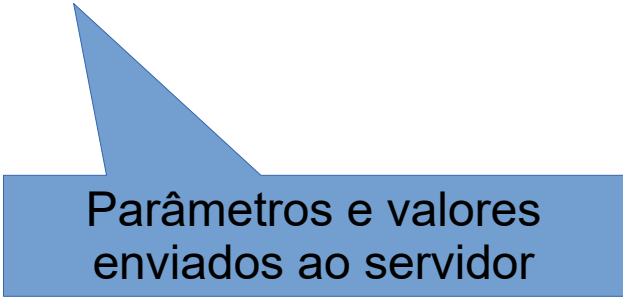


Parâmetro e valor enviado  
ao servidor



# Servidores HTTP - URLs

- `https://www.meusite.com?a=foo&b=bar`



Parâmetros e valores  
enviados ao servidor

# URL – Sintaxe completa

```
<scheme>://<user>:<password>@<host>:<port>/<path>?<query>#<frag>
```

**Scheme:** protocolo será usado para acessar o recurso. (HTTP, FTP, SMTP, etc).

- **Port:** A padrão é a 80 para http e a 443 para https
- **Query:** Utiliza o padrão nome=valor para envio de parâmetros
- **Frag:** Parte do recurso requisitado. Não é passado para o servidor. Geralmente é utilizado pelo navegador para encontrar uma parte do documento.

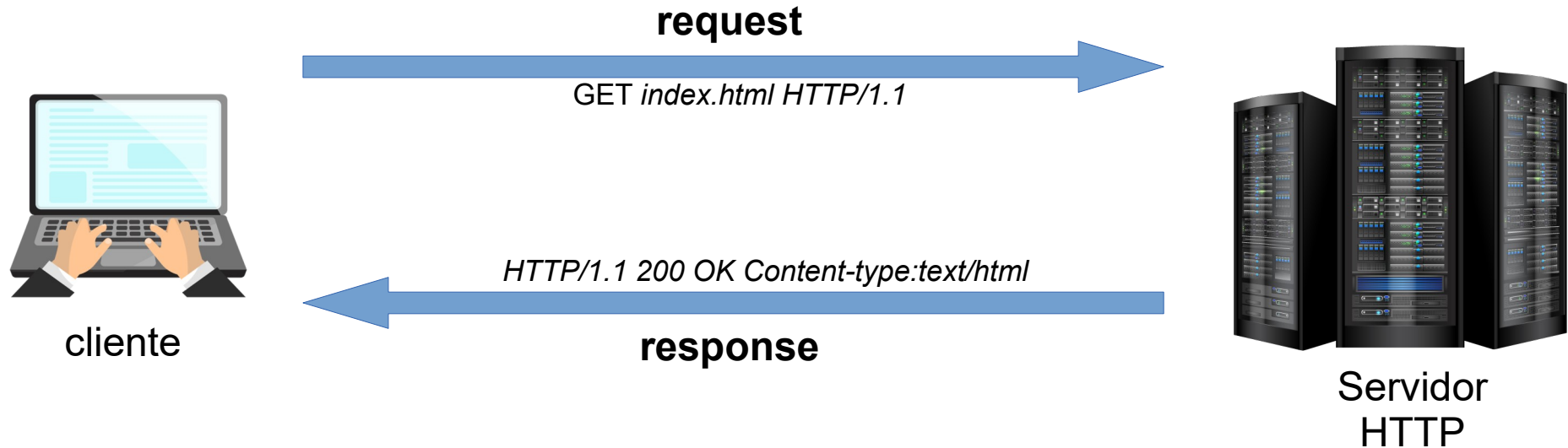
3/3

# Mensagens HTTP



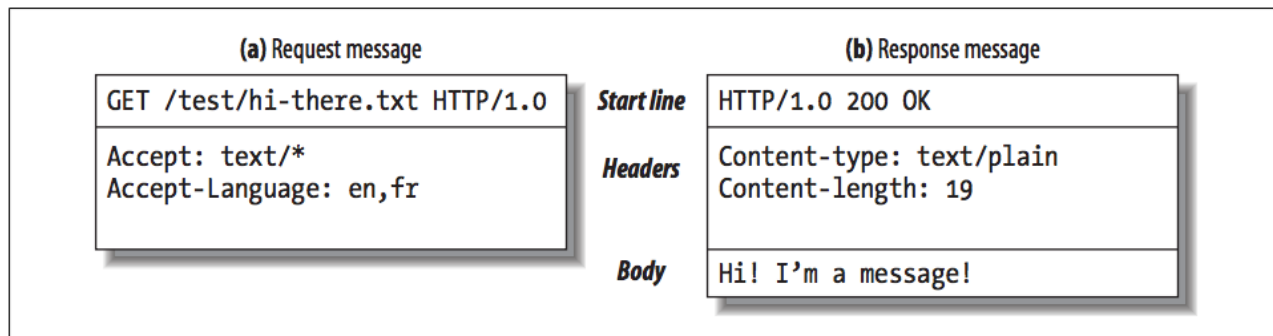
# Mensagens HTTP

Mensagens HTTP são os blocos de conteúdos trocados entre cliente e servidor



# Mensagens HTTP

- As mensagens de request ou response são blocos de texto formatados que possuem as partes:
  - Star line: diz ao servidor o que fazer
  - Headers: os atributos da mensagem
  - Body: contém os dados (opcional)



# Mensagens HTTP - request

- As mensagens de **request** possuem o formato ao lado
- Method: é a ação, ou o tipo de requisição. Pode ser:
  - **GET**
  - **POST**
  - PUT
  - DELETE
  - HEAD
  - OPTIONS
  - TRACE
  - ...
- Request-URL: o caminho do recurso desejado.
- Version: A versão do protocolo HTTP

```
<method> <request-URL> <version>  
  
<headers>  
  
<entity-body>
```

# Principais métodos

```
<method> <request-URL> <version>  
<headers>  
<entity-body>
```

MÉTODO	DESCRIÇÃO	MENSAGEM NO BODY?
GET	Obtém um recurso no servidor;	não
HEAD	Obtém somente os cabeçalhos de um recurso no servidor;	não
POST	Envia dados ao servidor para processamento;	sim
PUT	Armazena o conteúdo do Body de um request no servidor;	sim
TRACE	Rastreia as mensagens através dos servidores proxy até o servidor;	não
OPTIONS	Determina quais métodos podem ser manipulados no servidor;	não
DELETE	Remove um recurso do servidor.	não

# Mensagens HTTP - response

- As mensagens de **response** possuem o formato ao lado
- Status-code: é um número de três dígitos que descreve o que aconteceu durante a requisição. O primeiro dígito de cada código descreve o status geral, tais como sucesso, erro, etc.
- Reason-phrase: é uma versão mais legível para os humanos do código numérico do status-code;
- Headers: são as informações sobre os recursos disponíveis no servidor web, tais como data de criação, tamanho, tipo de conteúdo, etc.
- Entity-body: São os dados que a mensagem contém. Esses dados podem ser arquivos de imagem, som, vídeo, texto, etc. Nem todas as mensagens possuem este item.

`<version> <status-code> <reason-phrase>`

`<headers>`

`<entity-body>`



# Códigos de erro

```
<version> <status-code> <reason-phrase>  
<headers>  
<entity-body>
```

INTERVALO	CATEGORIA
100-199	Informações
200-299	Sucesso
300-399	Redirecionamento
400-499	Erro no cliente
500-599	Erro no servidor

# Códigos de erro frequentes

```
<version> <status-code> <reason-phrase>  
<headers>  
<entity-body>
```

INTER VALO	FRASE	DESCRIÇÃO
200	OK	Sucesso! O recurso requisitado se encontra no body da mensagem de resposta
401	UNAUTHORIZED	Você precisa de usuário e senha para acessar o recurso solicitado.
404	NOT FOUND	Recurso não encontrado
500	INTERNAL SERVER ERROR	Erro no servidor – geralmente bug no código server side
503	SERVICE UNAVAILABLE	Erro no servidor – geralmente sobrecarga

# Testando mensagens HTTP com Insomnia

<https://insomnia.rest/download>



# Testando seus conhecimentos



# Qual das opções abaixo não um um método http?

- PUT
- ADD
- GET
- DELETE
- POST

## Qual é a função do DNS?

- Localizar no disco um recurso.
- Autenticar os usuários.
- Traduzir um nome de domínio para o seu número de IP.
- Retornar o recurso requerido.
- Traduzir um número IP para o seu nome de domínio.

# Qual é a categoria do status code 404?

- Informações
- Sucesso
- Redirecionamento
- Erro no cliente
- Erro no servidor

# Qual é a finalidade do método **OPTIONS** do HTTP?

- Enviar dados para o servidor.
- Determinar quais métodos podem ser manipulados no servidor.
- Remover um recurso do servidor.
- Rastrear as mensagens através dos servidores proxies até o servidor.
- Obter um recurso do servidor.



# Qual é a categoria do status code 500?

- Informações
- Sucesso
- Redirecionamento
- Erro no cliente
- Erro no servidor

# Qual é a categoria do status code 200?

- Informações
- Sucesso
- Redirecionamento
- Erro no cliente
- Erro no servidor