



# AJAX

—

Prof. Victor Farias

v 1.2

# AJAX

- Com o AJAX é possível
  - Atualizar uma página web sem recarregar a página
  - Requisitar dados ao servidor
  - Receber dados do servidor
  - Enviar dados para servidor
- AJAX não é uma linguagem de programação
- AJAX = **A**synchronous **J**avaScript **A**nd **X**ML
- AJAX é uma combinação de
  - XMLHttpRequest, Fetch ou Axios para requisitar dado do servidor
  - JavaScript e HTML DOM para mostrar o dado
- Embora pareça que AJAX só pode transmitir XML, ele também pode transportar texto pleno ou JSON

# JSON

---

# JSON

- JSON é um estrutura de dados derivada da notação de objetos do JS
- Sintaxe:
  - Os dados estão dispostos em pares nome/valor
  - Os dados são separados por vírgula
  - Os objetos são colocados em chaves
  - Listas são colocadas entre colchetes

# JSON Exemplos

- Exemplo par nome/valor: `"firstName":"John"`
  - Nomes JSON requerem aspas duplas
  - Valores JSON podem ser números, strings, booleanos, listas, objetos ou null
- Exemplo objetos: `{"firstName":"John", "lastName":"Doe"}`
- Exemplo lista:

```
"employees":[  
  
  {"firstName":"John", "lastName":"Doe"},  
  
  {"firstName":"Anna", "lastName":"Smith"},  
  
  {"firstName":"Peter","lastName":"Jones"}  
  
]
```

# Protocollo HTTP



# Endereço IP e Portas

- **Endereço IP**

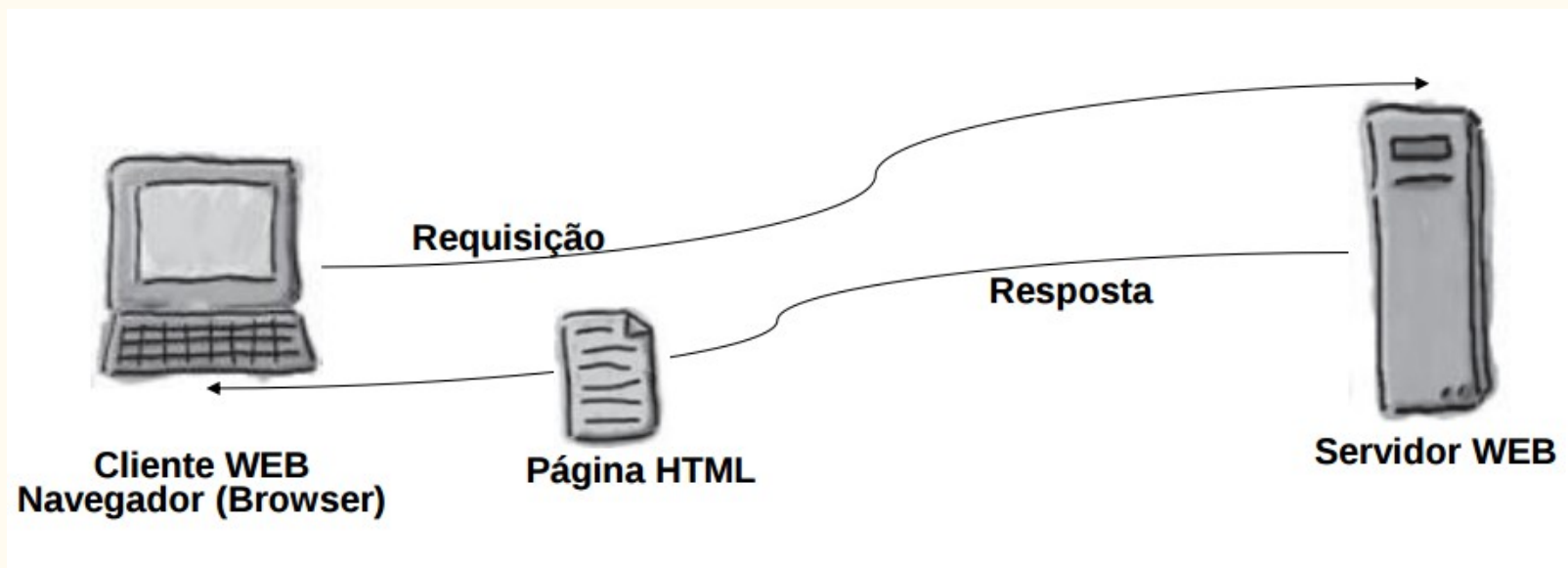
- Identifica um host na rede
- Cada interface de rede tem um IP
- ex: 200.21.32.43
- URLs são traduzidos em IP usando DNS (globo.com.br -> 186.192.90.5)

- **Portas**

- Identificam os processos de origem e fim
- Permite a comunicação de diversas aplicações na mesma máquina
- Cada aplicação recebe e envia requisições por uma porta
- ex: Servidor Web, por padrão, recebem requisições na porta 80

# Arquitetura Cliente Servidor

- **Servidor**
  - Aplicação que fornece serviço
  - Aceita requisições através da rede, em uma porta conhecida, e retorna o resultado
- **Cliente**
  - Processo que requisita um serviço
  - Para receber resposta, o cliente aloca uma porta arbitrária

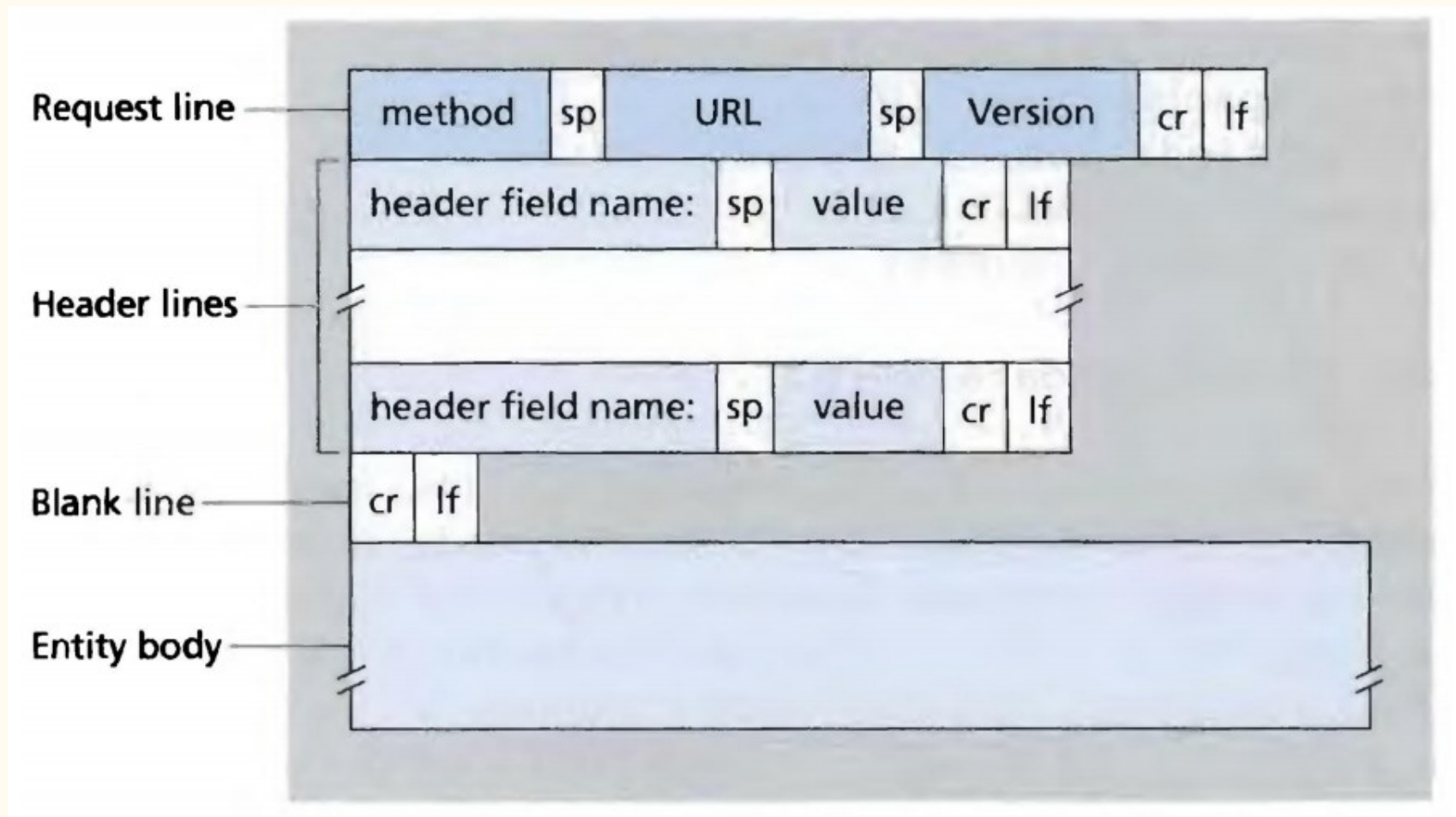




# Protocolo HTTP

- HTTP = **H**ypertext **T**ransfer **P**rotocol ou Protocolo de Transferência de Hipertexto
- Protocolo usado para transferir dados na WEB
- Funcionamento:
  - O cliente envia uma requisição HTTP para o servidor
  - O servidor envia uma resposta HTTP ao cliente

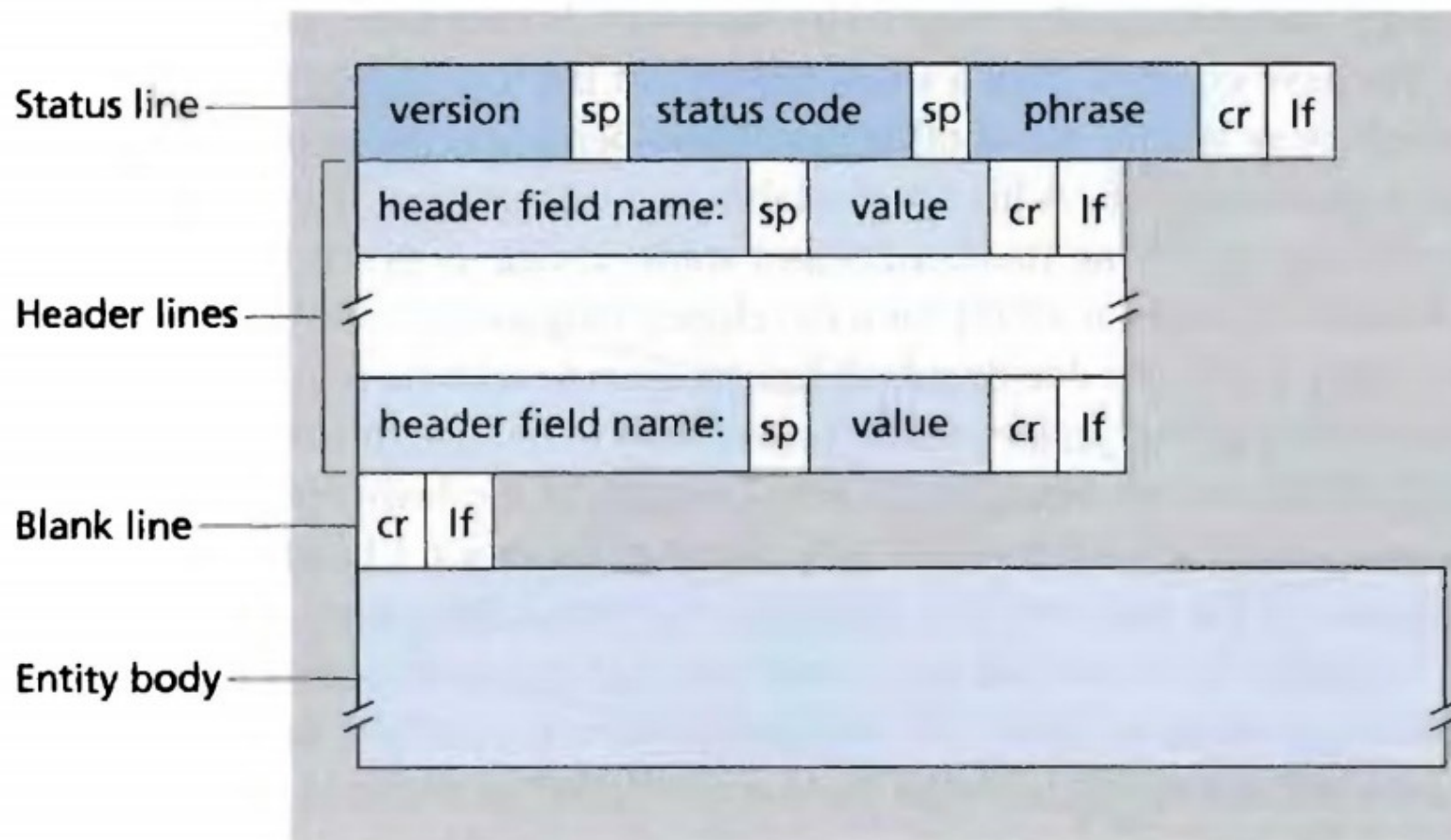
# Composição da Requisição HTTP



# Métodos de Requisição

- Métodos usados em requisições HTTP
  - GET – Solicita algum recurso
    - Dados são anexados à URL, ficando visíveis ao usuário
  - POST – Envia dados referentes ao recurso especificado para serem processados
    - Dados são incluídos no corpo do comando
  - PUT – Envia certo recurso
    - Em geral, é usado para atualizar dados
  - DELETE – Exclui o recurso
  - HEAD – Variação do GET em que o recurso não é retornado
    - Usado para obter metainformações por meio do cabeçalho da resposta, sem ter que recuperar todo o conteúdo.

# Resposta HTTP



# Código de Status

- O código de status é formado por três dígitos e o primeiro dígito representa a classe que pertence classificada em cinco tipos:
  - 1xx: Informational (Informação) – utilizada para enviar informações para o cliente de que sua requisição foi recebida e está sendo processada
  - 2xx: Success (Sucesso) – indica que a requisição do cliente foi bem sucedida
  - 3xx: Redirection (Redirecionamento) – informa a ação adicional que deve ser tomada para completar a requisição
  - 4xx: Client Error (Erro no cliente) – avisa que o cliente fez uma requisição que não pode ser atendida
  - 5xx: Server Error (Erro no servidor) – ocorreu um erro no servidor ao cumprir uma requisição válida
- O protocolo HTTP define somente alguns códigos em cada classe, mas cada servidor pode definir seus próprios códigos

# Axios



# Import Axios

```
<script src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></script>
```

# Sintaxe básica

```
axios({  
  method: "POST",  
  url: "https://jsonplaceholder.typicode.com/posts",  
  data: {nome: "Victor", sobrenome: "Farias"}  
});
```

Diagram illustrating the basic syntax of an axios request:

- Método** (Method): `method: "POST"`
- URL** (URL): `url: "https://jsonplaceholder.typicode.com/posts"`
- Body** (Body): `data: {nome: "Victor", sobrenome: "Farias"}`

## Formato do response:

```
{data: {...}, status: 200, statusText: "OK", headers: {...}, config: {...}, ...}
```



# Sintaxe básica

```
let request = async function(){  
  try{  
    let response = await axios({  
      method: "POST",  
      url: "https://jsonplaceholder.typicode.com/posts",  
      data: {nome: "Victor", sobrenome: "Farias"}  
    });  
  } catch (e){  
    console.log(e);  
  }  
}  
request();
```

Diagram labels:

- Método (points to "POST")
- URL (points to "https://jsonplaceholder.typicode.com/posts")
- Body (points to the data object)

**Regras async/await**

- O await deve ser utilizado para chamar funções async
- Só pode usar await dentro de uma função async

## Formato do response:

{data: {...}, status: 200, statusText: "OK", headers: {...}, config: {...}, ...}

# Padrão REST

---

# REST

- **REST = Representational State Transfer**
  - Abstração de arquitetura de um componente
  - Define forma de se consumir um dado recurso
  - Ignora implementação do componente
  - Foca na sintaxe de comunicação do componente
- **Usa protocolo HTTP**
  - Usa métodos/verbo (GET, POST, PUT ...) para indicar ação
  - URL define a estrutura do serviço
- **Usaremos JSON como formato de dados**

# Padrão REST - Verbos

- Verbos/métodos:
  - **GET**
    - Recuperar itens
  - **POST**
    - Adicionar item
  - **PUT**
    - Atualizar item
  - **DELETE**
    - Deletar item

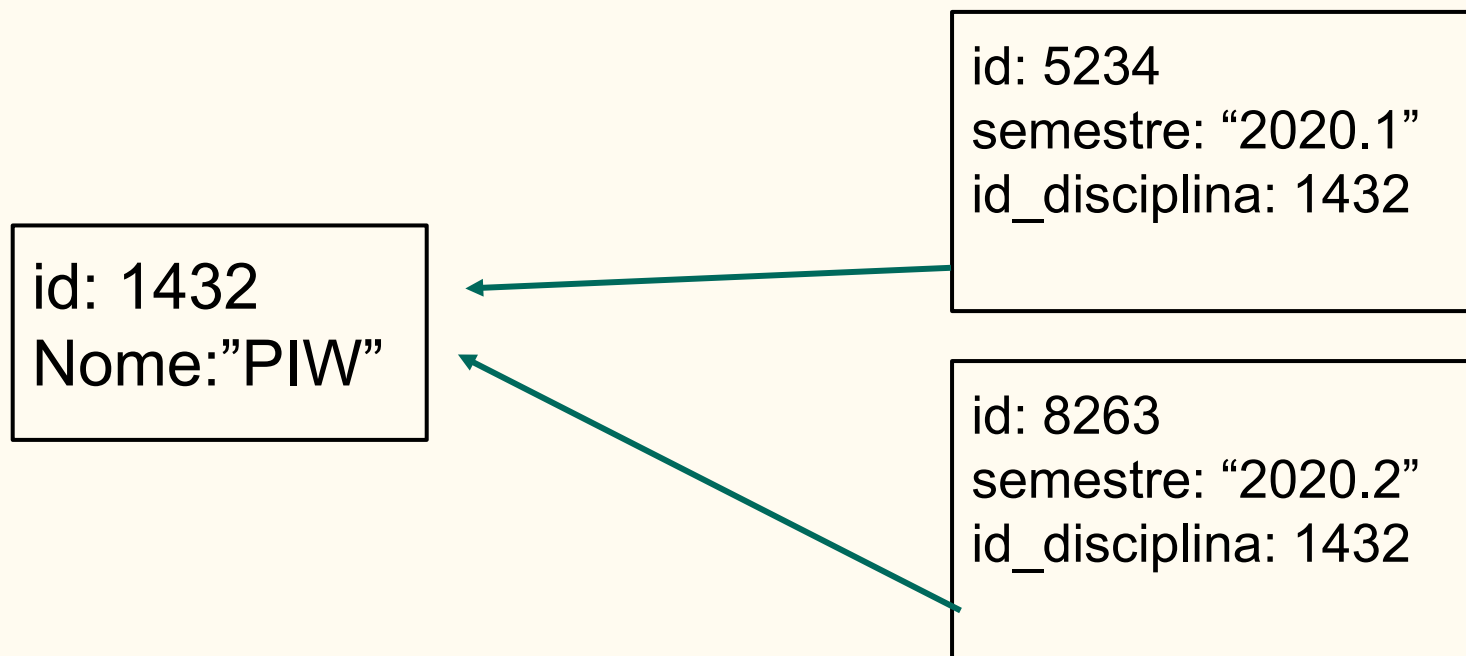
# Padrão REST - URL

- **GET /users**
  - Busca todos os usuários (retorna lista de usuários)
- **GET /users/1**
  - Busca o usuário com id 1
- **POST /users**
  - Insere um novo usuário
  - Novo usuário vai no corpo da requisição
- **DELETE /users/1**
  - Remove usuário com id 1

# Padrão REST - URL

- **GET    /users/1/comments**
  - Busca todos os comentários do usuários com id 1
- **GET    /comments?postId=1**
  - Busca todos os comentários cujo o id do post que ele foi feito é 1

# Relacionamento



# Perguntas?

Prof. Victor Farias