Requisições assíncronas.

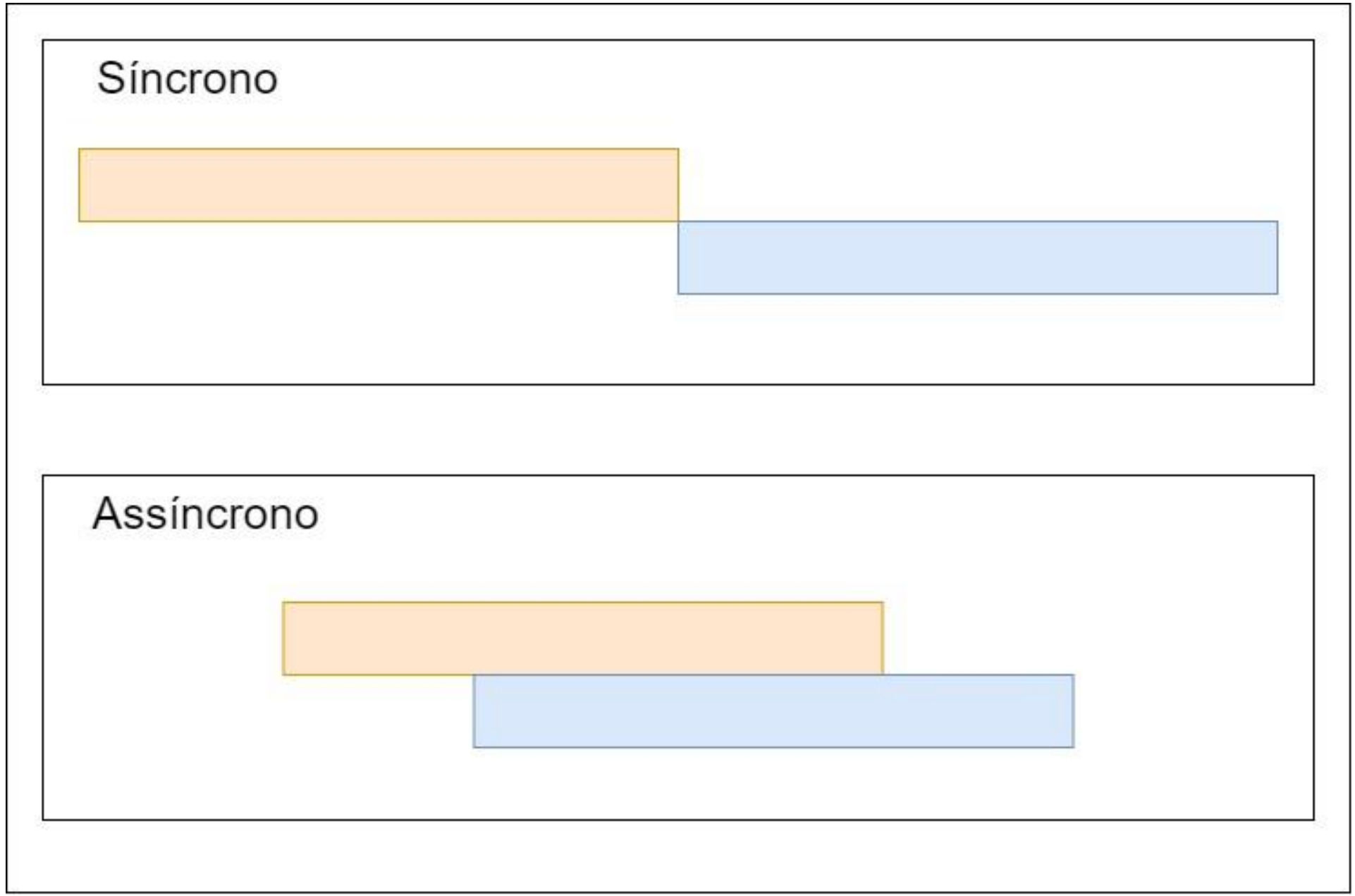
Prof. Me. Hélio Esperidião

Tipos de requisição: Síncrona

- Quando uma requisição é enviada, o processo remetente é bloqueado até que ocorra uma resposta, ou seja, não é possível enviar novas requisições até que nossa requisição atual seja finalizada, pois existe sincronismo entre as requisições

Tipos de requisição: Assíncrona

- Em uma requisição assíncrona, não existe sincronismo entre as requisições. Assim, podemos enviar diversas requisições em paralelo e cada resposta retorna quando estiver pronta


Síncrono
Assíncrono

Fetch

É uma interface JavaScript moderna para fazer requisições HTTP/HTTPS de forma assíncrona.

Essa API permite que os desenvolvedores criem aplicações web mais interativas e dinâmicas, oferecendo uma maneira mais intuitiva e fácil de realizar chamadas de rede.

Sintaxe

```
let uri ="endereço do serviço";
      fetch(uri, {
            method: "GET"
         }).then((response) => {
            return response.text();
         }).then((textoJson) => {
            //processar o texto json recebido da api
         }).catch((error) => {
            console.error("Error:", error);
         });
```

Exemplo 01

- Ao click de um botão fazer uma requisição para:
  - https://api.adviceslip.com/advice
- O serviço Retorna um json no formato:
  - {"slip": { "id": 163, "advice": "Big things have small beginnings."}}
- Escrever o resultado do JSON em um parágrafo.
- Funções que serão criadas:
  - carrecarDados()
  - processarDadosRecebidos(dados)

Exemplo 01 – Parte 1

```
let btn = document.getElementById("btnClick1");
   btn.onclick = function(){
      carrecarDados();
   }
```

Exemplo 01 – Parte 2

```
function  carrecarDados(){
      let uri ="https://api.adviceslip.com/advice";
      fetch(uri, {
            method: "GET"
         }).then((response) => {
            return response.text();
         }).then((textoJson) => {
            console.log(textoJson)
            processarDadosRecebidos(textoJson)
         }).catch((error) => {
            console.error("Error:", error);
         });
  }
```

Exemplo 01 – Parte 3

```
function processarDadosRecebidos(dados){
    let objJson = JSON.parse(dados);
    let id = objJson.slip.id;
    let advice = objJson.slip.advice;
    let divResposta = document.getElementById("divResposta");
    let p = document.createElement("p");
    let lblId = document.createTextNode("ID: " + id);
    let lblAdvice = document.createTextNode(" ADVICE: " + advice);
    p.appendChild(lblId);
    p.appendChild(lblAdvice);
    divResposta.appendChild(p);
}
```

Exemplo 02

- Ao click de um botão fazer uma requisição para:
  - https://parallelum.com.br/fipe/api/v1/carros/marcas
- O serviço Retorna um json no formato:
  - [{"codigo":"1","nome":"Acura"},{"codigo":"2","nome":"Agrale"},{"codigo":"3","nome":"Alfa Romeo"}]
- Escrever o resultado do JSON em um parágrafo.
- Funções que serão criadas:
  - carrecarDados()
  - processarDadosRecebidos(dados)

Exemplo 02 – Parte 1

```
let btn = document.getElementById("btnClick1");
    btn.onclick = function(){
        carrecarDados();
    }
```

Exemplo 02 – Parte 2

```
function carrecarDados() {
    let uri = "https://parallelum.com.br/fipe/api/v1/carros/marcas";
    fetch(uri, {
        method: "GET"
    }).then((response) => {
        return response.text();
    }).then((data) => {
        console.log(data)
        processarDadosRecebidos(data)
    }).catch((error) => {
        console.error("Error:", error);
    });
}
```

Exemplo 02 – Parte 3

```
function processarDadosRecebidos(dadosRecebidos) {
    let vetorObjetosJson = JSON.parse(dadosRecebidos);
    let divResposta = document.getElementById("divResposta");
    for (let objetoJson of vetorObjetosJson) {
        let codigo = objetoJson.codigo
        let nome = objetoJson.nome
        let p = document.createElement("p");
        let br = document.createElement("br");
        let lblCodigo = document.createTextNode("ID: " + codigo);
        let lblNome = document.createTextNode("MARCA: " + nome);
        p.appendChild(lblCodigo);
        p.appendChild(br);
        p.appendChild(lblNome);
        divResposta.appendChild(p);
    }
}
```

Exemplo CEP

- Ao click de um botão fazer uma requisição para:
  - https://viacep.com.br/ws/01001000/json/
  - O serviço Retorna um json no formato:


```
{
  "cep": "01001-000",
  "logradouro": "Praça da Sé",
  "complemento": "lado ímpar",
  "bairro": "Sé",
  "localidade": "São Paulo",
  "uf": "SP",
  "ibge": "3550308",
  "gia": "1004",
  "ddd": "11",
  "siafi": "7107"
}
```


Funções que serão criadas:

- buscarCep(cep)
- processarDadosRecebidos(dadosRecebidos)

Exemplo CEP

```
<input type="text" id="txtCep"><br>
<input type="text" id="txtLogradouro"><br>
<input type="text" id="txtBairro"><br>
<input type="text" id="txtLocalidade"><br>
<select id="cboEstado">
    <option value="AC">Acre</option>
    <option value="AL">Alagoas</option>
    <option value="AP">Amapá</option>
</select><br>
  <button id="btnClick1" >Buscar CEP</button>
```

Exemplo CEP

```
const btn = document.getElementById("btnClick1");
  btn.onclick = function(){
      const txtCep= document.getElementById("txtCep");
      let cep = txtCep.value;
      buscarCep(cep);
  }
```

Exemplo CEP

```
function buscarCep(cep) {
    let uri = "https://viacep.com.br/ws/"+cep+"/json/";
    fetch(uri, {
        method: "GET"
    }).then((response) => {
        return response.text();
    }).then((textoJson) => {
        console.log(textoJson)
        processarDadosRecebidos(textoJson)

    }).catch((error) => {
        console.error("Error:", error);
    });
}
```

Exemplo CEP

```
function processarDadosRecebidos(dadosRecebidos) {
    let objetoJson = JSON.parse(dadosRecebidos);
    const txtLogradouro= document.getElementById("txtLogradouro");
    const txtBairro= document.getElementById("txtBairro");
    const txtLocalidade= document.getElementById("txtLocalidade");
    const cboEstado= document.getElementById("cboEstado");
    txtLogradouro.value = objetoJson.logradouro;
    txtBairro.value = objetoJson.bairro;
    txtLocalidade.value = objetoJson.localidade;
    cboEstado.value = objetoJson.uf;

}
```