Requisições Assíncronas.

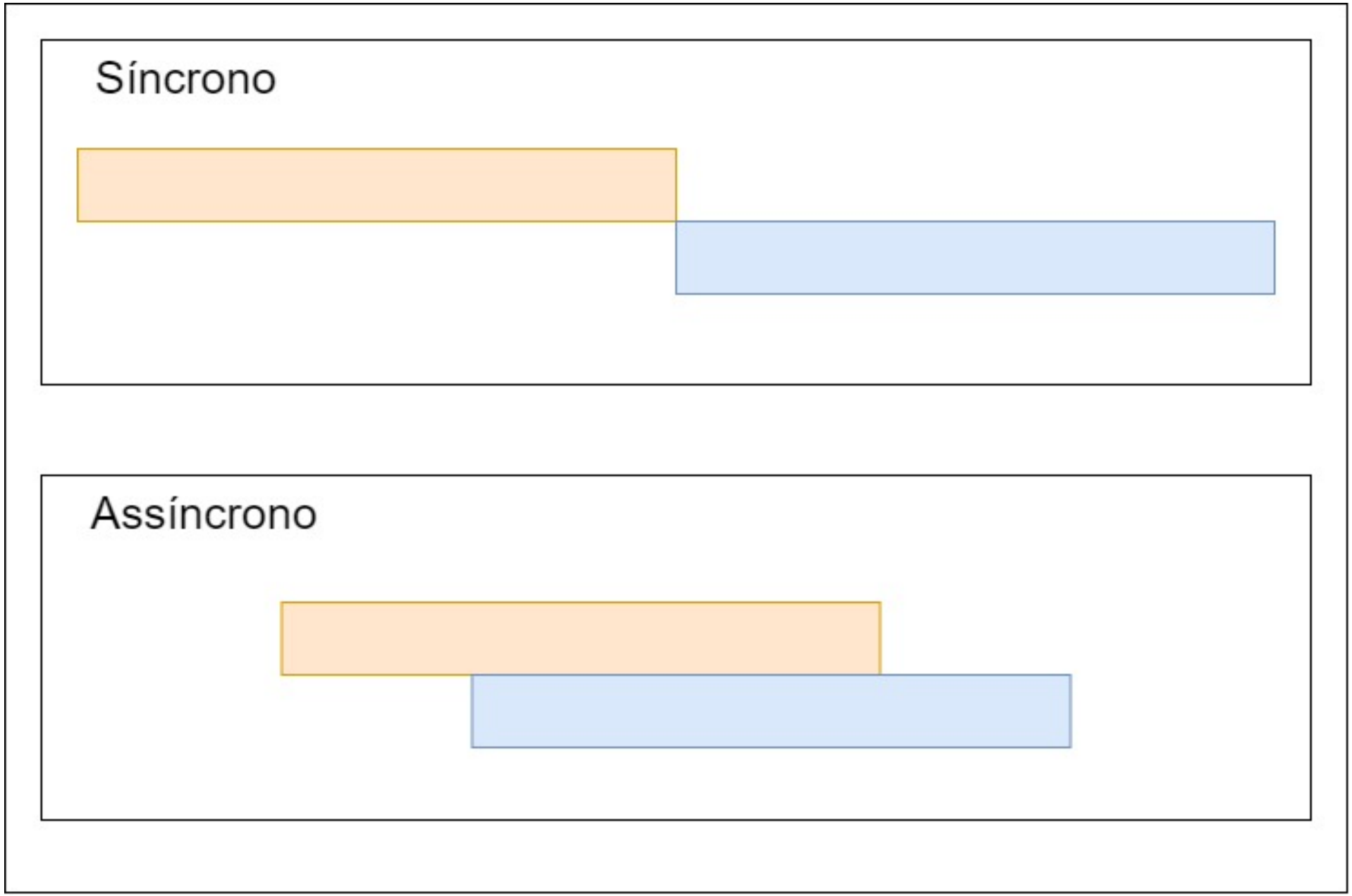
Prof. Me. Hélio Esperidião

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

```
// Função reutilizável que busca dados de uma URI fornecida
async function getDados(uri) {
  try {
    // Faz a requisição usando fetch
    const response = await fetch(uri);
    // Lê o corpo da resposta como texto
    const textoJson = await response.text();
    // Converte o texto em objeto JavaScript
    const objJson = JSON.parse(textoJson);
    // Exibe o JSON bruto no console (para depuração)
    console.log(textoJson);
    // Retorna o objeto convertido
    return objJson;
  } catch (error) {
    // Caso ocorra erro na requisição ou no parse, exibe no console
    console.error("Erro ao buscar dados:", 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)

```
<script>
 // Função reutilizável que busca dados de uma URI
fornecida
 async function getDados(uri) {
   try {
     // Faz a requisição usando fetch
     const response = await fetch(uri);
     // Lê o corpo da resposta como texto
     const textoJson = await response.text();
     // Converte o texto em objeto JavaScript
     const objJson = JSON.parse(textoJson);
     // Exibe o JSON bruto no console (para
depuração)
     console.log(textoJson);
     // Retorna o objeto convertido
     return objJson;
   } catch (error) {
     // Caso ocorra erro na requisição ou no
parse, exibe no console
     console.error("Erro ao buscar dados:",
error);
   }
 }
```

```
// Aguarda o carregamento do DOM antes de acessar elementos
document.addEventListener("DOMContentLoaded", () => {
  // Referência ao botão e à div de resposta
  const btnClick = document.getElementById("btnClick1");
  const divResposta = document.getElementById("divResposta");
  // Define a URL da API que será chamada
  const uri = "https://api.adviceslip.com/advice";
  // Adiciona evento de clique no botão
  btnClick.addEventListener("click", async () => {
    // Chama a função getDados passando a URI
    const objJson = await getDados(uri);
    // Se a resposta contiver o objeto esperado
    if (objJson && objJson.slip) {
      // Desestrutura o id e advice do objeto slip
      const { id, advice } = objJson.slip;
      // Adiciona o conselho ao conteúdo da div
      divResposta.innerHTML += `
            <p><strong>ID:</strong> ${id} <strong>ADVICE:</strong>
  ${advice}</p>
          `;
    } else {
      // Se não obteve conselho, informa o usuário
      divResposta.textContent = "Não foi possível obter o conselho.";
        }
      });
    });
  </script>
```

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)

```
<script>
    async function getDados(uri) {
        try {
            const response = await fetch(uri);
            const textoJson = await response.text();
            const objJson = JSON.parse(textoJson);
            return objJson;
        } catch (error) {
            console.error("Erro ao buscar dados:", error);
        }
    }
        document.addEventListener("DOMContentLoaded", () => {
        const btnClick = document.getElementById("btnClick1");
        const divResposta = document.getElementById("divResposta");
        const uri = "https://parallelum.com.br/fipe/api/v1/carros/marcas";
        btnClick.addEventListener("click", async () => {
            divResposta.textContent = "Carregando...";
            const objJson = await getDados(uri);
            divResposta.textContent = ""; // Limpa antes de exibir
            objJson.forEach(({ codigo, nome }) => {
                divResposta.innerHTML += `<p><strong>ID:</strong> ${codigo} <strong>MARCA:</strong> ${nome}</p>`;
            });
        });
    });
</script>
```

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>
<div id = “msg”> </div>
```

```
document.addEventListener("DOMContentLoaded", () => {
      const btn = document.getElementById("btnClick1");
      const msg = document.getElementById("msg");
      btn.addEventListener("click", async () => {
      const cep = document.getElementById("txtCep").value.trim();
      if (!cep) {
          exibirMsg("Por favor, digite um CEP válido."); return;
        }
      const uri = `https://viacep.com.br/ws/${cep}/json/`;
      exibirMsg("Buscando informações do CEP...");
      const dados = await getDados(uri);
      if (!dados) {
          exibirMsg("Erro ao buscar CEP. Tente novamente.");
          limparCampos();return;
      } else if (dados.erro) {
          exibirMsg("CEP não encontrado.");
          limparCampos();return;
     }
    preencherCampos(dados);
    exibirMsg("CEP encontrado com sucesso!");
 });
```

```
function preencherCampos(dados) {
 document.getElementById("txtLogradouro").value = dados.logradouro || "";
 document.getElementById("txtBairro").value = dados.bairro || "";
 document.getElementById("txtLocalidade").value = dados.localidade || "";
 document.getElementById("cboEstado").value = dados.uf || "";
}

  function limparCampos() {
       document.getElementById("txtLogradouro").value = "";
       document.getElementById("txtBairro").value = "";
       document.getElementById("txtLocalidade").value = "";
       document.getElementById("cboEstado").value = "";
     }

  function exibirMsg(texto) {
       msg.textContent = texto;
     }
   });
```