

Título: Parte Teórica — Consumo de APIs e DOM Manipulation

Aluno: Leonardo dos santos da silva — UniFECAF — Análise e Desenvolvimento de Sistemas

Disciplina: Web Programming for Front End

1. O que é uma API e como funciona o consumo de dados em tempo real

Uma API (Application Programming Interface) é um conjunto de regras que permite a comunicação entre aplicações diferentes. No contexto web, APIs REST expõem endpoints HTTP que retornam dados em formato JSON. O consumo em “tempo real” consiste em o cliente (navegador) fazer requisições periódicas ou sob demanda (por eventos) para obter dados atualizados e exibi-los. No front-end isso é feito com `fetch` ou bibliotecas (axios), tratando Promises para processar a resposta e atualizar o DOM.

2. Conceito de DOM manipulation e criação de elementos dinâmicos via JavaScript

O DOM (Document Object Model) representa a árvore de elementos HTML de uma página. Manipular o DOM permite alterar a interface sem recarregar a página. Para criar conteúdo dinamicamente usamos `document.createElement(tag)`, configuramos atributos (ex.: `setAttribute`, `className`) e inserimos no documento com `appendChild` ou `append`. Essa técnica é essencial quando os dados vêm de uma API e os elementos precisam ser gerados conforme o retorno.

3. Funções básicas utilizadas para consumo da API (`fetch`, `then`, `createElement`, `appendChild`, etc.)

- `fetch(url)`: inicia requisição HTTP e retorna uma `Promise` com um `Response`.
- `response.json()`: converte o corpo da resposta para objeto JavaScript (`Promise`).
- `await / then`: mecanismos para trabalhar com Promises.
- `document.createElement('div')`: cria um elemento.
- `element.appendChild(node)`: adiciona um nó filho ao elemento.
- `element.textContent`: define o texto de um elemento.

No projeto eu usei `fetch` com `async/await`, e criei cards usando `createElement + appendChild` para garantir que o HTML seja gerado somente após os dados chegarem.

4. Motivo da escolha da API (Jikan API (MyAnimeList))

Escolhi a Jikan API (MyAnimeList) por ser pública, estável, oferecer imagens e metadados (informações gerais dos personagens), e por não exigir token de autenticação — o que facilita a demonstração prática sem configurações extras. A API também possui paginação, o que me permitiu demonstrar controle de navegação (anterior/próxima).

5. Regras de acesso e documentação da API utilizada

Endpoint principal: `https://api.jikan.moe/v4/anime/5114/characters`

Resposta: JSON com campos `info` (página, total) e `results` (lista de personagens).

Não há necessidade de token; por ser pública, a API pode aplicar rate limits para evitar abuso — por isso meu código trata erros e evita muitas requisições simultâneas.