DW

MDM

2022/2023
1º semestre

Ex. 2 — Web APIs

Inês Pereira
2019210452

Link Projeto: https://github.com/

API escolhida

Para a elaboração deste trabalho, optei por utilizar a API do Art Institute of Chicago, que me permite apresentar várias das suas obras de arte públicas de modo dinâmico.

Optei por a escolher por me permitir aceder e apresentar dados de vários tipos — texto e imagem —, bem como responder aos prossupostos deste trabalho:

- Apresentação geral dos resultados
- Pesquisa através de formulário e pedido específico à
- Filtragem dos resultados apresentados do lado do cliente
- Ordenação dos resultados apresentados

Link: https://api.artic.edu/api/v1/artworks/ → (específico a obras de arte)
https://api.artic.edu/ → Apresentação API

Descrição das funcionalidades implementadas

Apresentação geral dos resultados da API

Obras de arte de vários tipos (categorias).
Cada “cartão” corresponde a uma obra e contempla:

- o seu nome e ano de execução (quando foi concluída)
- o nome do artista
- o tipo de obra (categoria: pintura, impressão, fotografia...)
- imagem da mesma

Deste modo, focou-se o conteúdo mostrado no essencial, sendo que tinha como objetivo a exposição das várias obras, achando que não fazia sentido conter muito mais informação, como um texto descritivo. Aqui, procurava a apresentação das obras como se o “cliente” circulasse pelo museu, só procura mais informação da obra, caso lhe seja despertado o interesse.

Pesquisa através de formulário e pedido específico à API

Neste site, é possível pesquisar por nome da obra, sendo que a informação recebida da chamada à API é filtrada a cada adicionar de caracter no campo de pesquisa — não se tem que clicar em nenhum botão para receber os dados pretendidos, estes são apresentados de modo dinâmico.

Foi necessário recorrer a um EventListener para que, sempre que o campo de pesquisa recebesse alterações, se atualizasse os dados, apagando os atuais e desenhando os filtrados (que incluíssem o input do cliente).

Ao se ter uma variável que recebe o texto inserido, converte-se em maiúsculas para que, ao se pesquisar por obras que contivessem ou correspondencem à pesquisa, se pudesse receber todos os dados possíveis e não só os que contivessem exatamente o inserido.

Exemplo da importância da uniformização em maiúsculas:

Sem **.toUpperCase()**
ex.: Pesquisa → '**sa**' Dados apresentados: 'Portrait of El**sa** Glaser',

Com **.toUpperCase()**

'Portrait of El**sa** Glaser'
'**Sa**lome with the Head of **Sa**int John the Baptist'
...

Explicação:

Inicialmente, a filtragem dos dados da chamada à API resulta somente nos resultados que contenham exatamente '**sa**' no nome, ignorando os que começam com letra maiúscula (**'Sa'**).

Filtragem dos resultados apresentados do lado do cliente

A filtragem é feita através das **categorias das obras**.

Como não se sabe inicialmente as obras que vão calhar na pesquisa, optei por gerar os elementos correspondentes às categorias de forma dinâmica, no JavaScript, sendo que, independentemente de serem apresentadas obras de uma só categoria ou mais, estes elementos vão corresponder sempre aos dados recebidos. Esta gestão é feita através do atributo “artwork_type_title”. Caso este equivalha ao valor da categoria selecionada (pelo uso do botão a tal destinado), os dados serão filtrados consoante a mesma.

Um dos desafios que existiram ao desenvolver o código foi a existência de categorias com várias palavras. Aparecia um erro relacionado com os espaços (“html”) das categorias por associar classes com os nomes das mesmas a cada elemento.

Para o resolver, optei por trocar todos os espaços por “_”, sendo que deste modo, passou a ser reconhecido como uma só palavra/classe.

Ordenação dos resultados apresentados

A ordenação é feita através dos anos de finalização das obras (por **ordem cronológica**), sendo que é possível ordenar por ordem crescente e/ou decrescente.

O código referente a este ponto, foi adaptado do código disponibilizado pelo professor.